LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

AD-A277 270

DTIC ACCESSION NUMBER

LEVEL

(1)

INVENTORY

ASC-TR-94-5009

DOCUMENT IDENTIFICATION

Aug 93

Approved for public release
Distribution Unlimited

DISTRIBUTION STATEMENT

HANDLE WITH CARE

ACCESSION FOR

NTIS GRA&I ☑
DTIC TRAC ☐
UNANNOUNCED ☐
JUSTIFICATION

BY

DISTRIBUTION/

AVAILABILITY CODES

DISTRIBUTION	AVAILABILITY AND/OR SPECIAL	
A-1		

DISTRIBUTION STAMP

DTIC
ELECTE
MAR 1 0 1994
S C D

DATE ACCESSIONED

DATE RETURNED

94 3 15 007

DATE RECEIVED IN DTIC

474P 94-08403

REGISTERED OR CERTIFIED NUMBER

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

DTIC FORM JUN 90 70A

DOCUMENT PROCESSING SHEET

PREVIOUS EDITIONS MAY BE USED UNTIL STOCK IS EXHAUSTED.

LOAN DOCUMENT

ASC-TR-94-5009

MODULAR SIMULATOR SYSTEM (MSS) INTERFACE DESIGN DOCUMENT FOR THE GENERIC MSS

K KELLY, J BROWN,
G KAMSICKAS, W TUCKER

BOEING DEFENSE AND SPACE GROUP
SIMULATION AND TRAINING SYSTEMS
499 BOEING BLVD
HUNTSVILLE, AL 35824

AUGUST 1993

FINAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

SYSTEMS ENGINEERING DIVISION
AERONAUTICAL SYSTEMS CENTER
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7126

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

JEFFREY C. VALITON, Maj, USAF
Program Manager
Special Programs Divsion

James D Basinger

JAMES D. BASINGER
Team Leader
Special Programs Division

JAMES J. O'CONNELL
Chief, Systems Engineering Division
Training Systems Program Office

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify ASC/YTSD, WPAFB, OH 45433-7111 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE 20 Aug 93

3. REPORT TYPE AND DATES COVERED FINAL

4. TITLE AND SUBTITLE
Modular Simulator System (MSS)
Interface Design Document for the Generic MSS

5. FUNDING NUMBERS
F33657-86-C-0149
64227F

6. AUTHOR(S) R. Kelly, J. Brown
G. Kamsickas, W. Tucker

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Boeing Defense and Space Group
Simulation and Training Systems
499 Boeing Blvd
Huntsville, AL 35824

8. PERFORMING ORGANIZATION REPORT NUMBER
S495-10735-1

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Aeronautical Systems Center
Systems Engineering Division
Bldg 11 2240 B St Ste 7
Wright-Patterson AFB, OH 45433-7111

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ASC-TR-94-5009

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release;
distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This document describes the detailed design for the interfaces associated with the Modular Simulator System. Tailoring will be necessary to meet specific training system requirements. Specific tailoring instructions are included for each paragraph. It is suggested that the user read the "Modular Simulator Engineering Design Guide" and the "Modular Simulator Management Guide" prior to tailoring this document.

14. SUBJECT TERMS
Modular Simulator System (MSS)

15. NUMBER OF PAGES 426

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev 2-89)
Prescribed by ANSI Std Z39-18
298-102

TABLE OF CONTENTS

APPENDICES

LIST OF FIGURES

Preface

This generic Modular Simulator System (MSS) Interface Design Document (IDD) has been developed in accordance with DI-MCCR-80027A, Data Item Description for Interface Design Document. This document is designed to be tailored for a specific aircraft training device or family of aircraft training devices. Training devices may consist of Weapon System Trainers (WST), Operational Flight Trainers (OFT), Cockpit Procedures Trainers (CPT), Part Task Trainers (PTT), etc.

Tailoring will be necessary to meet specific training system requirements. The tailoring should be accomplished so as not to violate the goals and intent of the MSS concept. It is assumed that users of this document have a familiarity with MSS design concepts and architecture, and general working knowledge of aircraft training systems. It is suggested that the user read the "Modular Simulator Engineering Design Guide" (D495-10440-1) and the "Modular Simulator Management Guide" (D495-10439-1) prior to tailoring this specification. These guides provide an overview of the MSS architecture, in-depth discussion of its application, and lessons learned from previous applications.

Each segment in the MSS architecture provides a portion of the overall system functionality. Similar functions and operations were grouped in each segment based on past experience, areas of design expertise, and management of inter-segment communication. To promote reuse of the segments and gain the maximum benefits of using the MSS approach, it is suggested that the user adhere to the generic functional allocation. Interfaces between segments should remain relatively constant from application to application. The application vehicle is considered to be military aircraft (e.g. fixed wing, variable geometry or rotary wing), although the MSS concepts and architecture can also be applied to ground and sea vehicles.

This specification contains specific tailoring instructions for each paragraph. The instructions are contained within the paragraphs, and are identified by blank spaces and/or italicized instructions. When the tailoring process is complete, the italicized tailoring instructions should have been replaced by the application specific text or deleted from the specification. Paragraphs which do not apply to a particular application should not be deleted. They should be identified as "Not Applicable" to maintain paragraph numbering consistency.

1. SCOPE

1.1 Identification. This Interface Design Document (IDD) describes the detailed design for the interfaces associated with the ***(insert application aircraft)*** Modular Simulator System (MSS). This document will focus on the design of interfaces between ***(insert application aircraft)*** MSS segments. Specifically those used for communication of data via the MSS Virtual Network (VNET).

(This paragraph should be tailored to identify the specific interfaces that this document will address. The intent of this document is to define the generic MSS inter-segment VNET interfaces. However, when this document is used in an actual application it should be used to define all system level interfaces. This would include interfaces to hardware and software shared by segments. The MSS architecture does not define or specify the internal design of the segments. If this is required by an application, it is suggested that a separate IDD be created at the segment level to define those interfaces. When additional system level interfaces are added to this document they should begin with paragraph 3.3 and follow the preparation instructions provided in DI-MCCR-80027A, Data Item Description for Interface Design Document.)

1.2 System Overview. The MSS defines a generic, standard architecture for a training simulator. The architecture consists of an interface scheme, a partitioning scheme, and an allocation of requirements to the various partitioned components. This document specifies the interface design for the ***(insert application aircraft)*** MSS. Figure 1.2-1 illustrates the fundamental partitioning of the ***(insert application aircraft)*** MSS. Individual segments communicate with each other via the MSS Virtual Network (VNET). The VNET communication architecture is a conceptual mechanism using a message passing protocol and independent of the hardware implementation. The goal of the VNET is to provide a generic communication architecture that is adaptable to both high and low end applications while accommodating computer technology advances.

For the ***(insert application aircraft)*** MSS, the following segments are necessary:

a. Flight Station (FS)

b. Flight Controls (FC)

c. Flight Dynamics (FD)

d. Propulsion (PRO)

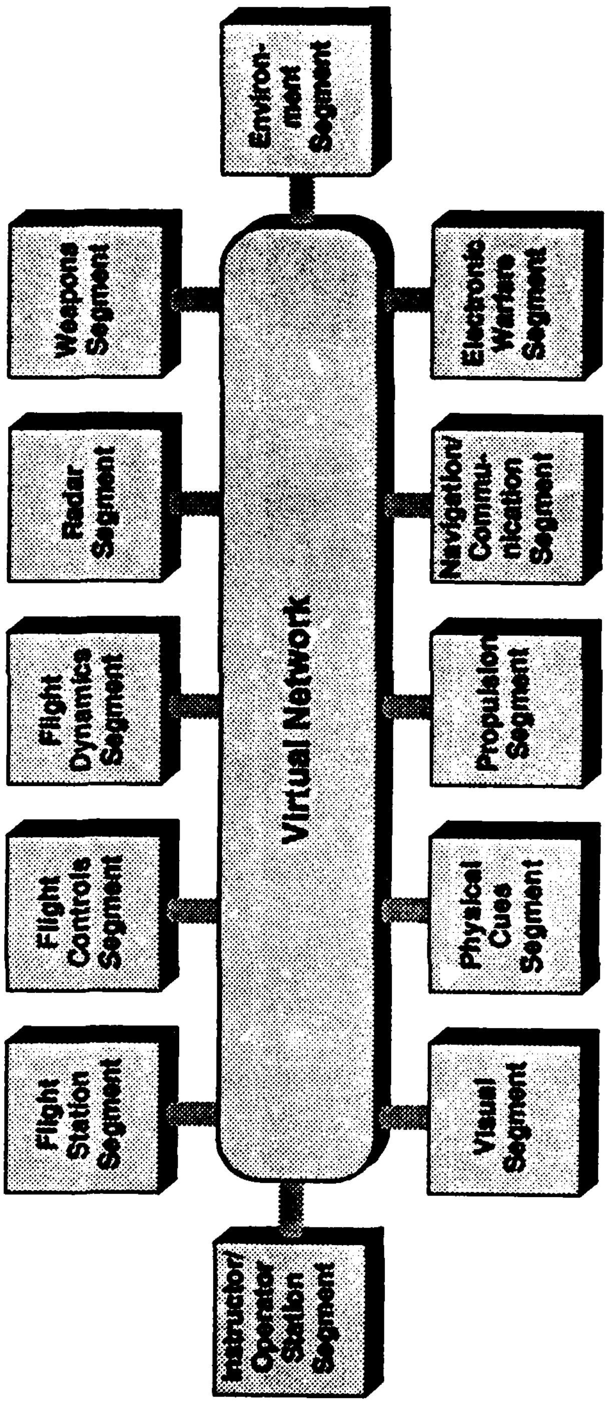

Figure 1.2-1 Fundamental MSS Partitioning

e. Navigation/Communication (NAV)

f. Weapons (WPN)

g. Radar (RDR)

h. Electronic Warfare (EW)

i. Physical Cues (PHC)

j. Visual (VIS)

k. Instructor Operator Station (IOS)

l. Environment (ENV)

(The list of MSS segments should be tailored to correspond with the requirements of this application aircraft training device. The fundamental partitioning for the MSS consists of twelve unique segments. Any one or more of these segments may be combined within a single computational system. In addition, Figure 1.2-1 must be tailored to reflect the application aircraft top level architecture. Finally, this paragraph should be tailored to provide a unique description of the system.)

1.3 Document Overview. This IDD establishes the interface design applied to the development of the ***(insert application aircraft)*** MSS. This IDD is the companion document to the MSS System/Segment Specification (SSS) and the MSS Interface Requirements Specification (IRS). It describes the detailed design of the interfaces between the ***(insert application aircraft)*** MSS segments. This document is used for four purposes:

a. To describe and present the detail design of the interfaces between segments.

b. To be used by MSS personnel as the basis for detailed software design of the inter segment interfaces.

c. To communicate and control interface design decisions to the customer.

d. To provide the customer a means for assessing compliance with interface design requirements.

This document was prepared in accordance with MSS design standards. The Notes section lists abbreviations and acronyms used in this document. This IDD conforms to the requirements of DOD-STD-2167A and the format specified Data Item Description DI-MCCR-80027A.

2. REFERENCED DOCUMENTS

2.1 Government Documents. The following documents of the exact issue shown form a part of this specification to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this specification, the contents of this specification shall be the superseding requirement.

SPECIFICATIONS:

Federal - ***(Identify applicable federal specifications)***

Military - ***(Identify applicable military specifications)***

Other Government Agency - ***(Identify applicable government specifications)***

STANDARDS:

Federal - ***(Identify applicable federal standards)***

Military - ***(Identify applicable military standards)***

Other Government Agency - ***(Identify applicable government standards)***

DRAWINGS:

(Identify applicable drawings)

OTHER PUBLICATIONS:

Manuals - ***(Identify applicable manuals)***

Regulations - ***(Identify applicable regulations)***

Handbooks - ***(Identify applicable handbooks)***

Bulletins - ***(Identify applicable bulletins)***

Copies of specifications, standards, drawings, and publications required by suppliers in connection with specified procurement functions should be obtained from the contracting agency or as directed by the contracting officer.

(In this paragraph list only those documents which are explicitly referenced within this specification. If a requirements paragraph is tailored to reference a System/Segment Specification Volume paragraph, and that paragraph contains a referenced document,

list it here. All requirements and references in the System Specification Volume I are requirements of this specification unless specifically excluded in this document.)

2.2 Non-Government Documents. The following documents of the exact issue shown form a part of this specification to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this specification, the contents of this specification shall be the superseding requirement.

SPECIFICATIONS:

(Identify applicable non-government specifications)

STANDARDS:

(Identify applicable non-government standards)

DRAWINGS:

(Identify applicable non-government drawings)

OTHER PUBLICATIONS:

(Identify additional, applicable non-government publications)

Technical society and technical association specifications and standards are generally available for reference from libraries. They are also distributed among technical groups and using Federal Agencies.

(In this paragraph list only those documents which are explicitly referenced within this specification. If a requirements paragraph is tailored to reference a System/Segment Specification Volume paragraph, and that paragraph contains a referenced document, list it here. All requirements and references in the System Specification Volume I are requirements of this specification unless specifically excluded in this document.)

3. INTERFACE DESIGN

Detailed design for the interfaces defined in this document have been derived from the interface requirements established by the ***(insert application aircraft)*** MSS IRS.

The following paragraphs provide the external interface design for the inter segment interfaces in the ***(insert application aircraft)*** MSS.

3.1 Interface Diagrams. The ***(insert application aircraft)*** MSS is decomposed as illustrated in Figure 3.1-1. This decomposition follows the segment partitioning as required by the IRS.

(This paragraph and Figure 3.1-1 should be tailored to provide a description of the interfaces at a system level. Figure 3.1-1 should graphically illustrate all system level interfaces between segments, allocation of segments to computational systems, implementation of the VNET, and backdoor interfaces used by segments. Additional diagrams may be added to convey more detail regarding specific interfaces. The allocation of segments to computational elements is important in a MSS. The segments may all be in one module, one segment per module, or some combination thereof. How many segments allocated to a module is a systems engineering decision based on system requirements. Segments should not be aware of their allocation. This will make segment software more portable to other hardware platforms. VNET communication is between segments, not modules. The VNET implementation must ensure that the method/mechanism of communication is transparent to the segments beyond the segment's interface to the VNET.)

Figure 3.1-1 illustrates the allocation of segments to modules. A MSS module represents a computational element to which one or more segments are allocated. Segments communicate via the VNET, which is transparent to individual segments. Individual segments are unaware of their module allocation.

3.2 MSS Inter-Segment Interface. The ***(insert application aircraft)*** MSS architecture requires all data flows outside the segment to take place over the VNET in the form of messages defined in Appendix A of this IDD. All MSS segments transfer these messages using the same VNET services regardless of their computational system allocation. The underlying mechanism for communicating between segments is invisible to the segment and handled by the VNET.

3.2.1 Data Elements. The data elements required for the design of the ***(insert application aircraft)*** MSS interface are defined in Appendix A of this IDD. No specific data element definition table or cross-reference is provided. Appendix A consists of compilable Ada code, therefore, all available CASE tools may be utilized to identify and record desired information. The following paragraphs discuss the

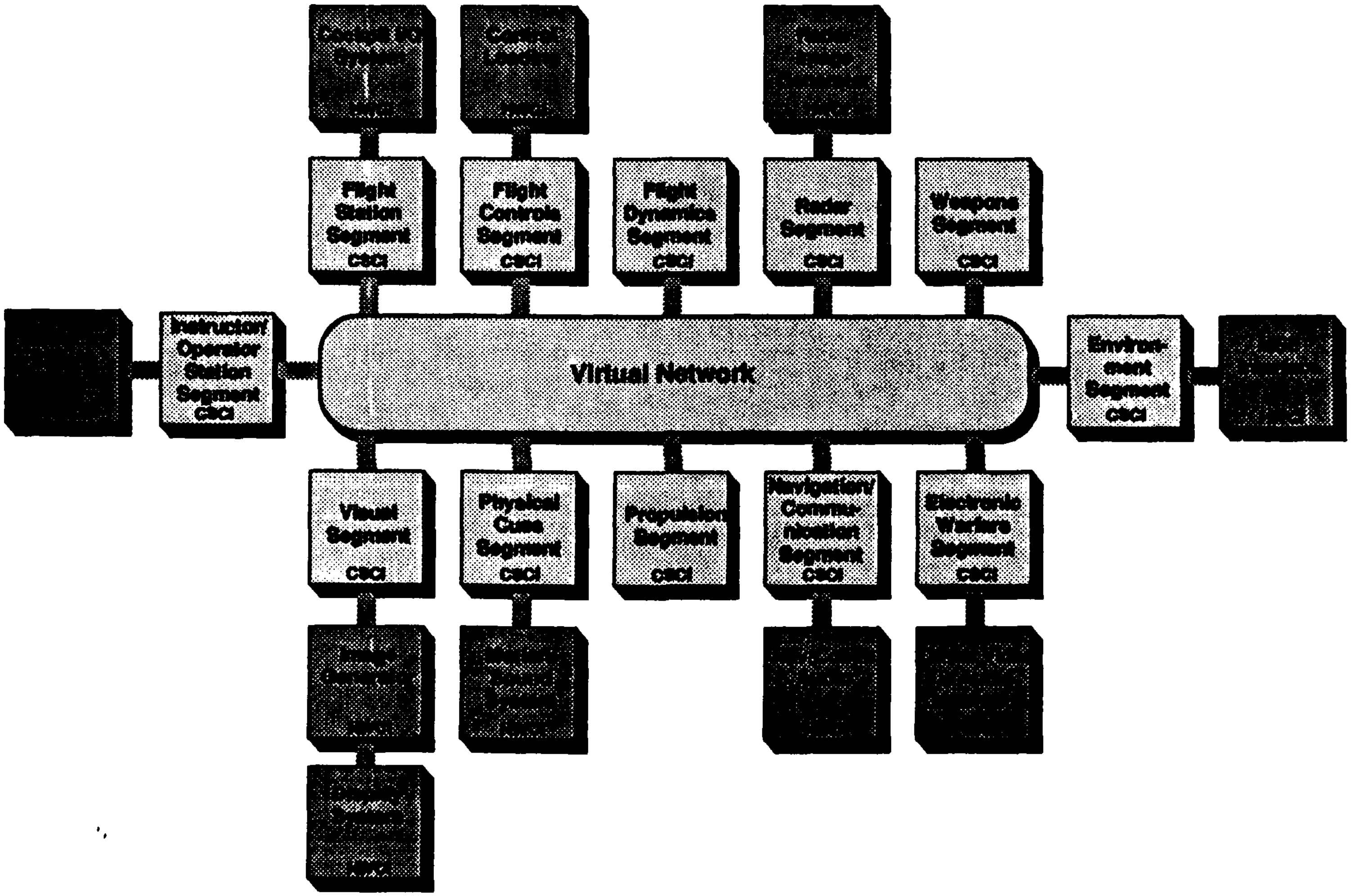

Figure 3.1-1 MSS Decomposition

underlying representation and organization of the MSS inter-segment interface data elements.

3.2.1.1 Data Element Organization. The data elements are defined in the ***(insert application aircraft)*** MSS through the use of Ada types packages. The requirements basis for the types packages can be found in the ***(insert application aircraft)*** MSS IRS. These types packages are illustrated in Figure 3.2.1.1-1. The types packages are divided into two general categories; global types packages and segment specific interface types packages. The global types packages are used to define data types that are used by all segments at a system level. Objects declared from these types are the basis for all inter-segment communication. Segment specific interface types packages are used to define data types that are specific or unique to a segment. These packages are also used to define the project unique identifiers for each inter-segment interface. Data types defined in these packages use the global data types to define the complex data types used for inter-segment communication.

There are nine different types packages used to define the MSS inter-segment interface. Each types package builds on the previous package as shown in Figure 3.2.1.1-1. In other words, the segment specific interface types packages contain types (records, arrays, etc.) whose fields are based on types declared in the global types packages.

3.2.1.1.1 Primitive Types. The Primitive_Types package is used to isolate the machine dependency of Ada types to a single package. This types package contains all the specific declarations of types for use in the MSS. Example types include; Float_64, Float_32, Integer_32, Unsigned_Integer_16, and Integer_8. The declarations include precision and resolution for all numeric types. The primitive types package also contains the representation specifications that adequately describe and establish the underlying method of machine storage for data of these primitive types. The Primitive_Types package is inherited or "withed" by only the Base_Types package. Direct reference to it by any of the segment interface types is not allowed.

3.2.1.1.2 Base Types. The Base_Types package is the types package that is used by all other global types packages and segments to replace the use of package Standard in the Ada library. Types defined in the Base_Types package are extensions of the primitive types. This package also includes the declaration of the types for Discrete_State and Sim_Boolean. The purpose of this package is to remove machine dependencies from the inter-segment interfaces.

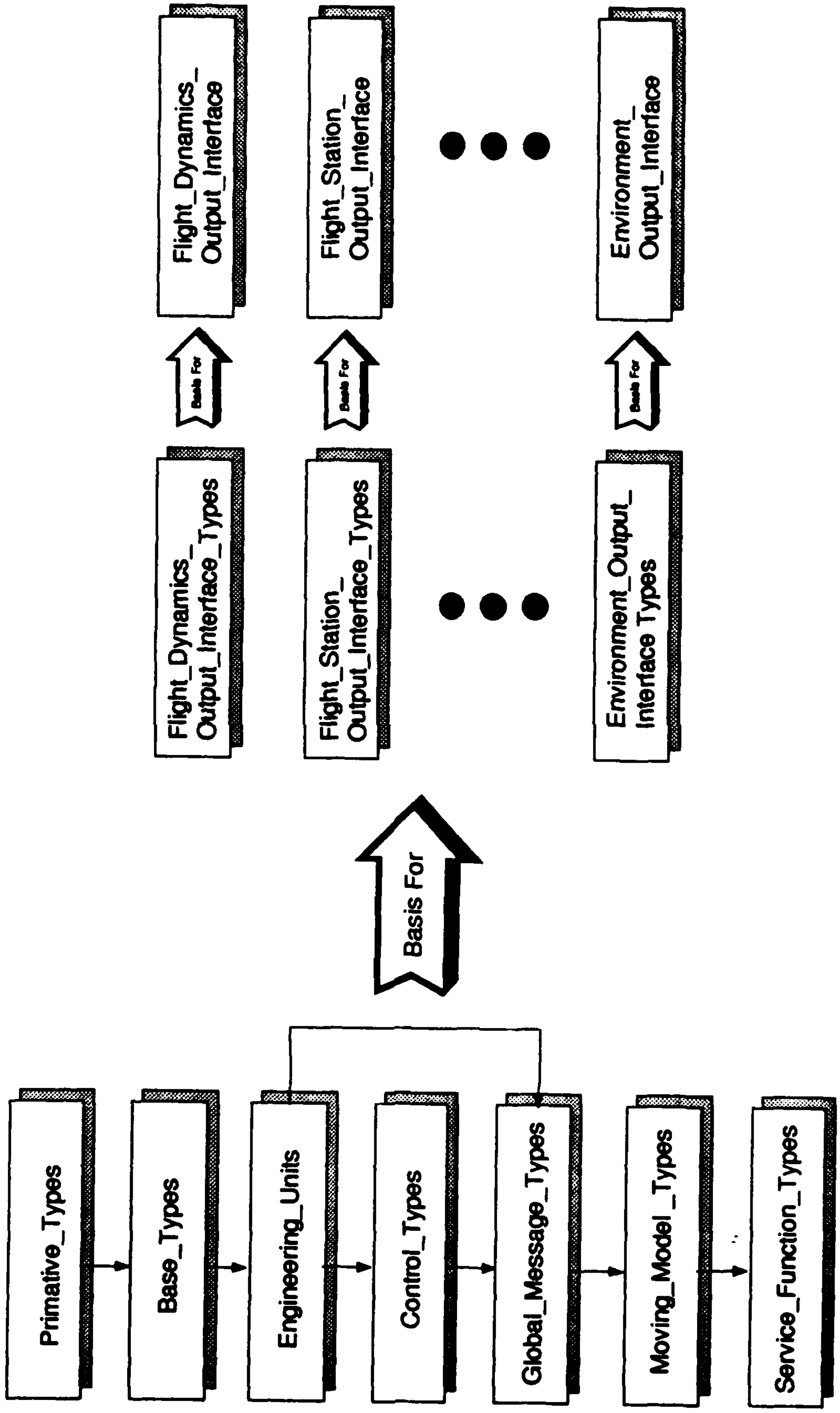

Figure 3.2.1.1-1 MSS Message Data Build Process

3.2.1.1.3 Engineering Units. The Engineering_Units type package contains all global engineering units and constants. These types include declarations for accelerations, velocities, lengths, and various coordinates system types. The engineering units are used to constrain interface data to specific units. By providing the units as an integral part of the interface the user of the data is always aware of the basis for the data and can perform conversions accordingly.

3.2.1.1.4 Control Types. The Control_Types package contains the specific control types for the *(insert application aircraft)* MSS device. The control types are used at a system level by the segments for system control, by the VNET for interface management, and by test and diagnostic tools for system maintenance. Example types in this package include the enumerations for subsystems and components and the types for system mode and state control.

(This paragraph should not require tailoring. However, the Control_Types package will require tailoring for the particular simulation. The enumerations dealing with subsystems, components, and their associated subtypes need to be tailored to the specific application. The change will consist of deleting enumeration values.)

3.2.1.1.5 Global Message Types. The Global_Message_Types package contains aircraft/simulator types that are used by all segments. The package contains aircraft configuration type definitions, simulator system control types, and common module type definitions. Any type that is shared or duplicated between segment messages is found in this package. This package provides the enumeration types that define the unique characteristics of the *(insert application aircraft)* aircraft. Common data structures that define basic aircraft components and systems are also contained in this package. These data structures are reusable from application to application.

(This paragraph should not require tailoring. However, the contents of this package will require tailoring between simulators. The aircraft configuration types will require tailoring for each application to define the aircraft's unique configuration. The enumeration types should be revised to reflect aircraft equipment and characteristics.)

3.2.1.1.6 Moving Model Types. The Moving_Model_Types package contains the definition and limiting constants for simulated moving models. The package contains all data representations for moving model data types including the enumerated list of possible moving models. Moving models include companion aircraft, threat platforms, chaff, flares, weapons, etc.

(This paragraph should not require tailoring. However, this package will require some tailoring if moving models are required in the application simulation. This package may be deleted from the interface if there are no moving model requirements for the

application. The text for this paragraph should be replaced with "Not Applicable" if moving models are not required.)

3.2.1.1.7 Service Function Types. The Service_Function_Types package contains the data types for the MSS service functions. The MSS service functions are MSS functions that may be performed by several segments.

(This paragraph should not require tailoring. However, this package will require some tailoring to allocate service functions and their parameters as required in the application simulation. This package may be deleted from the interface if there are no service function requirements for the application. The text for this paragraph should be replaced with "Not Applicable" if service functions are not required.)

3.2.1.1.8 Segment Specific Output Interface Types. All segment specific data elements are captured in types packages named for each segment. The format for the package name is: "<segment name>_Output_Interface_Types". These packages specify types for messages which are output only by their respective segment. The output interface types are the basis for the complex data elements used to create the inter-segment interfaces and are based on the types found in the global types packages. Each segment's output interface package contains four sections; Aircraft/Simulator Specific Segment Types, Aircraft/Simulator Reusable Segment Types, Segment Output Records, and Segment Representation Specs. These sections provide for simple maintenance and tailoring of the package contents. Appendix A contains one output interface types package for each segment. These packages contain the detailed design information that is required by the IDD data element definition table.

(This paragraph requires no tailoring. However, the sections of Appendix A containing the aircraft/simulator specific types, must be modified to match the requirements of the aircraft being simulated or the requirements for the simulator. As a general rule, the contents of the section containing reusable types will not need to be modified. The representation specs in the private part are designed to require little or no modification.)

3.2.1.1.9 Segment Specific Output Interfaces. There is a corresponding output interface package for each output interface types package. This package contains the declarations for each of the segment's output messages. The format for the package name is: "<segment name>_Output_Interface". These packages contain the exact messages used for inter-segment communication including the project unique identifier for each data element. Appendix A contains one output interface package for each segment.

(This paragraph should not requiring tailoring. The first step in adapting the code is to determine which of the functions in this segment will not be performed, based on simulator requirements. The messages associated with these functions need not be sent, and should therefore be deleted from the package or commented out. These packages will take the brunt of change between each simulator. The types are just the

templates and even if not used do not require deletion or modification. The Messages on the other hand will need to be tailored to the particular application.

Each message declaration is followed by a comment line containing "Destination:" and the abbreviations of the segment(s) which receive this message. These comments should be modified to account for (a) the presence or absence of the other segments, and (b) the requirements of the other segments for data. For example, if the EW segment is absent, then the notation that the EW segment is a destination of a given message should be removed.)

3.2.1.2 Data Element Information. The information contained in the following paragraphs provides a correlation between the data required for a data element definition table and appendix A. The interfaces defined in Appendix A contain the information required by the data element definition table. Figure 3.2.1.2-1 illustrates how a typical MSS data element provides the required information.

3.2.1.2.1 Project Unique Identifier. Each MSS message or data element that is transmitted via the VNET has a unique identifier. All messages used on the VNET are declared as Ada objects in the segment specific output interface packages. As shown in Figure 3.2.1.2-1, Atmosphere_Quarter_Rate_Output is the project unique identifier for data type Atmosphere_Quarter_Rate.

3.2.1.2.2 Data Element Description. There are no textual descriptions for each data element. Each data element is defined using the Ada software language. The use of Ada allows the actual code to provide a brief description of the data element. Comments are included in appendix A where required to provide clarification.

3.2.1.2.3 Data Element Source. Data elements are grouped into Ada packages by data source. The only data sources in the MSS are the segments. The data source for each data element is included in the package name declaration as shown in Figure 3.2.1.2-1.

3.2.1.2.4 Data Element User. The data element user for each message is defined in the "Destination" comment for each data element. This is illustrated in Figure 3.2.1.2-1. The segment destination for each user is defined using the standard set of MSS segment acronyms.

3.2.1.2.5 Data Element Units of Measure. The majority of data elements are defined using the English system of units. All engineering units for the interface are defined in the Engineering_Units type package. Additional units can be added as required, but it is suggested that the existing units set be used to promote reuse and reduce compatibility problems. Figure 3.2.1.2-1 identify typical units of measure for one data element.

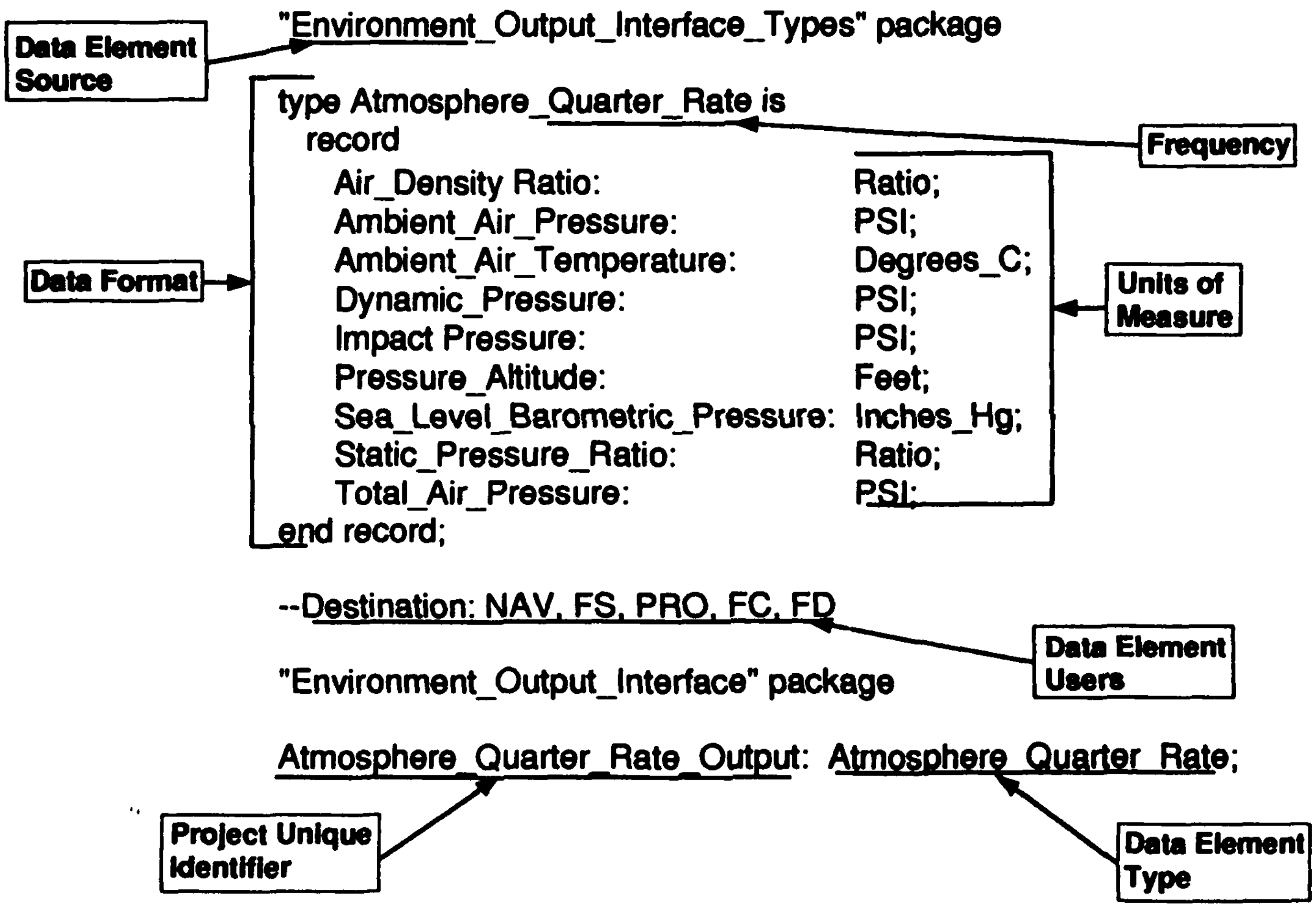

Figure 3.2.1.2-1 Data Element Information

3.2.1.2.6 Data Element Limit/Range of Values. Because of the generic quality of the MSS inter-segment interface, it is virtually impossible to define limits and ranges for all data elements. The Appendix A interface does impose limits on some data types, such as normalized numbers and degrees, where possible.

3.2.1.2.7 Data Element Accuracy. There are no specific accuracy requirements for inter-segment data. Therefore, data element accuracy is not defined beyond the precision of numeric data types.

3.2.1.2.8 Data Element Precision. Data element precision and resolution are defined in the Primative_Types and Base_Types packages in Appendix A. All other type definitions for numeric data use the definitions in these two packages. This allows the MSS interface to isolate machine dependencies and improve portability to any hardware platform.

3.2.1.2.9 Data Element Frequency. There are two types of data element refresh rates in the MSS, synchronous and asynchronous. Asynchronous data elements are called "send-on-change" data elements and are identified as such in Appendix A. Synchronous data elements have the update rate identified in the data element type and project unique identifier as shown in Figure 3.2.1.2-1. The Maximum, or "max" rate for the ***(insert application aircraft)*** MSS is ***(insert maximum iteration rate)*** Hz as defined by the MSS IRS. All other rates are a division of the maximum rate.

3.2.1.2.10 Data Element Legality Checks. Data legality checks are performed by the Ada compiler for the MSS. The interfaces provided in Appendix A are fully compilable, MIL-STD-1815A, Ada software. This ensures that all segments correctly interpret the inter segment interface. The legal transmission of data elements by the VNET is handled by the VNET interface.

3.2.1.2.11 Data Element Data Type. All data elements are defined using strict Ada data typing. As shown in Figure 3.2.1.2-1, many of the data elements are complex data types, such as Ada records. The data elements that are used to construct these complex types also use strict data typing to ensure an unambiguous interface definition.

3.2.1.2.12 Data Element Data Representation. All MSS data elements have a specific data format. Ada software constructs are used to define the format of data elements. To improve portability of these data formats, each data element is further defined by Ada Representation Specifications. The Representation Specifications are provided in Appendix A as an integral part of the inter-segment interfaces.

3.2.1.2.13 Data Element Priority. Synchronous data elements have priority over asynchronous data elements. All synchronous data elements are transmitted at the specified update rate. The clock tick message, originating from the IOS segment, provides the system level synchronization. This message has the highest priority among the data elements.

3.2.2 Messages Descriptions. The ***(insert application aircraft)*** MSS interface messages can be broken down into the following categories: system operation, system management, inter-model data transfer, and inter-model state transfer. The specific interface messages nomenclature, organized by segment, is located in the ***(insert application aircraft)*** MSS IRS. The specific interface message content is provided in Appendix A of this IDD. The previous discussion on the structure and organization of the data elements serves as a cross-reference mechanism for the interface messages.

3.2.3 Interface Priority. The highest priority interface message in the ***(insert application aircraft)*** MSS is the system clock message, which originates from the IOS segment. The next priority interface messages are the synchronous messages originating from each segment. The lowest priority interface messages are the asynchronous, "send-on-change" messages also originating from each segment. The priority classification for messages is given in the message definition. The final issue related to interface priority is the frame of origination for each message. Appendix B of this IDD records all of the messages' frame of origination.

3.2.4 Communication Protocol. The communication protocol defines the actual software and hardware mechanisms for transmission of information via the VNET. The following paragraphs describe the VNET design and the structure of the messages.

3.2.4.1 Virtual Network Design. The ***(insert application aircraft)*** MSS VNET provides the physical and data interface between segments. The VNET utilizes the Xpress Transfer Protocol (XTP) as defined by Protocol Engines Inc. (PEI) specification 89-103 Revision 3.5 and the Fiber Distributed Data Interface (FDDI) physical media as defined by the American National Standards Institute (ANSI) x3.166-1989, FDDI - Physical Layer Medium Dependent specification. The ***(insert application aircraft)*** MSS VNET provides the functional capabilities to (a) read messages from the network addressed to a particular segment; (b) strip from the message the header and trailer data not needed by the segment application layer; (c) reformat the data to a form compatible with the segment's computational system; and (d) reverse these processes for messages transmitted by the

segment, including multiple addressee messages. The VNET insures that the interface is reliable and includes error and exception handling. All message data is formatted as specified in Appendix A of this IDD and paragraph 3.2.4.2.

(This paragraph must be tailored to indicate the actual physical media and data interface used in this application aircraft MSS. This paragraph must also be tailored to identify the communication protocol utilized for this application aircraft. The choice of FDDI/XTP is not binding, although it has been demonstrated. The selection of a VNET implementation will be driven by the selection of system requirements. Since the actual network is, or should be, of no concern to the segment, a suitable name for it is "Virtual" Network.

The VNET interface design described in the following paragraphs is provided as an example. Paragraph 3.2.4.1 and all sub-paragraphs should be replaced with the actual VNET design for the application. The FDDI/XTP based VNET implementation should be used as a default since it is a proven solution. Other implementations should consider message traffic and reuse potential in their design. The tailoring instructions in the following paragraphs provide design guidance for VNET implementations.)

3.2.4.1.1 Virtual Network Structure. The ***(insert application aircraft)*** MSS VNET provides the means for Applications to communicate with one another. Applications are insulated from the VNET through an interface called Application Services. Figure 3.2.4.2.1-1 illustrates that the VNET is comprised of four major divisions; segment application, application services, VNET interface, and the VNET media. The segment application layer contains the simulation models that support the functional capabilities allocated to the segment. Application layers communicate with other application layers through the VNET via a predefined set of application layer communication services. The ***(insert Application aircraft)*** MSS VNET's segment application layer is separate from the VNET, but interfaces to the VNET through the application services layer to a VNET interface. The logical interface definitions for the data being transmitted and received by the segment application layers are defined in Appendix A, of this IDD.

The VNET interface provides the communication link to and from the VNET media. The VNET interface layer is referred as a "session" layer in some protocols. The ***(insert Application aircraft)*** MSS VNET interface utilizes the non-proprietary XTP protocol.

The VNET media provides the actual transfer of information between different segments as they are represented by their individual VNET interfaces. The VNET media layer is referred to in some protocols as a "transport" layer. The ***(insert Application aircraft)*** MSS VNET media layer utilizes a FDDI token ring.

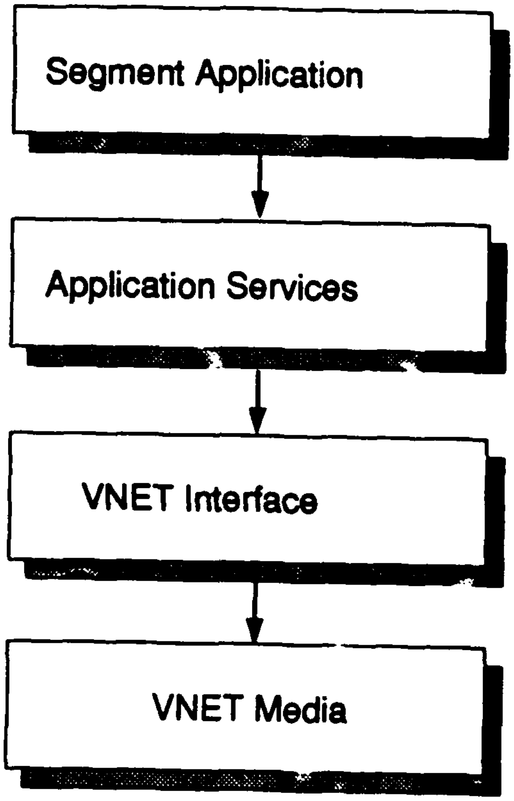

Figure 3.2.4.1.1-1 MSS VNET Structure

(The structural layers implied in these paragraphs are essential to the MSS concept; therefore, no tailoring of these layers is permitted. An alternative, non-proprietary protocol may be specified, provided it meets the Modular Simulator requirements.)

3.2.4.1.2 VNET Application Services Design. The following pseudo-code outlines the VNET's Application Services design by function.

a. Get function:
   1. Makes data available to the Application
   2. VNET Interface does not modify Application data except during a Get call.
   3. Returns one of the following as a status:
      i. No data ever
      ii. No new data
      iii. New data
      iv. Illegal receiver for this message
   4. Makes only complete copies of a given message available to the Application
   5. Returns the number of messages provided

b. Put function:
   1. Sends data to the VNET
   2. Data in transit to the VNET Interface is protected from writes by the Application
   3. Automatic, invisible retransmission on error
   4. Notifies Application when catastrophe prevents sending data
   5. Returns one of the following as a status:
      i. OK
      ii. Illegal sender for this message

c. I_Am function:
   1. Identifies segment to VNET Interface

d. Message list:
   1. Identifies messages, legal senders, legal receivers

e. Keep_Copies function:
   1. Identifies n[illegible]er of copies to be kept by VNET Interface

f. Notify_Of_Arrival [illegible]nction:
   1. Application says to VNET: when this message arrives, execute this code at once.

(This paragraph, and pseudo-code, must be tailored to reflect the actual VNET design utilized for the application aircraft MSS. Additional administrative functions or services may be added as long as they do not interfere with or obstruct the current services. Some services may be deleted if not required for the application. Services should be added or deleted using the same style and format as the current services.)

3.2.4.1.3 VNET Internal Data Structures. The following paragraphs describe the general behavior and characteristics of the *(insert application aircraft)* MSS VNET design.

3.2.4.1.3.1 Internal Identifiers. The *(insert application aircraft)* MSS VNET utilizes a set of internal identifiers. The VNET identifies each message with an integer code. Likewise, the VNET identifies each segment with an integer code. This design approach enhances the VNET's ability to map messages to the legal senders and receivers, and for segments to inform the VNET about their status. Derivation of the message identifiers is provided in paragraph 3.2.4.2.1.

3.2.4.1.3.2 Internal Objects. The *(insert application aircraft)* MSS VNET declares an object for every interface message. These objects correspond to the objects identified in the Appendix A segment output interface packages. Comment fields near the message object declarations in these packages name all the other segments to which the message can legally be sent. An off-line CASE tool constructs a legal sender and receiver table, embedded in the runtime VNET code, that controls message transmission.

(This paragraph must be tailored to indicate the actual practice utilized to protect mis-transfer of messages in this application aircraft MSS. There are other acceptable approaches to resolving this problem, for example, one could place the message names and the legal sender and receiver information into a data file which all the VNET Interfaces read. The Message Identifiers would then correspond to the line numbers in the data file on which the messages appear.)

3.2.4.1.3.3 Message Buffers. The *(insert application aircraft)* MSS VNET requires a number of buffers for storing the interface message stacks, for access by the application. These buffers are located on the VNET processor. One implication of this design is that the VNET Interface needs only to return the address of the buffer containing new data to the Application to make the transfer.

(This paragraph must be tailored to reflect the actual buffering design utilized in this application aircraft MSS. The design presented has limitations -- it does not support multi-copied messages or support multiple computational systems accessing the same VNET Interface. A trade study of bus traffic and buffer management indicated that an efficient design would move all the copies of a message from the VNET to the Application's memory on a Get call. Then, when the Application wants to get the next copy of the message, the Application Services would simply advance the pointer without polling the VNET.)

3.2.4.1.3.4 Message Memory. The *(insert application aircraft)* MSS VNET requires enough memory to support the memory space required for message traffic in a segment. Since each segment has different numbers and sizes of messages, the required amount of memory space will vary from segment to

segment. The VNET provides memory space to handle the message traffic for the segment with the highest memory requirements in the system to allow for reuse of the VNET interface among segments. Memory within the VNET is provided for the VNET software, XTP, FDDI board drivers and any applicable VNET support programs.

(This paragraph must be tailored to indicate the actual memory requirements for the application aircraft MSS. This requirement ensures that the same VNET will be used in all segments. This is normally the most efficient method of design. However, if each segment is to have a unique VNET the memory requirement should be based on other considerations such as spare capacity and future expansion.

The designer should complete a trade study to ensure that the number of messages times the number of copies that are to be received will fit within the allocated memory. This may impact the decision about where to locate the message buffers for the Applications. For example, having 11 segments in the same computational system, each with 100 copies of a 1000-byte message might require keeping the messages in the VNET Interface space.)

3.2.4.1.4 VNET Processing. Because the ***(insert application aircraft)*** MSS VNET software must receive messages in the background (i.e., while the Application is running), the VNET software must contain a process which does this receiving. That is, the VNET Interface may not simply be a subprogram which executes as part of the segment Application. The ***(insert application aircraft)*** MSS uses a separate CPU in the segment computational element for the VNET process.

(This paragraph must be tailored to indicate the design solution for the application aircraft MSS design. Alternative, acceptable designs exist.)

3.2.4.1.4.1 Message Interrupts. In the ***(insert application aircraft)*** MSS, Applications interrupt the VNET software for every message request. Simple multiplication of the number of interrupts required, by latency time for each interrupt, revealed that responding to a large number of interrupts will tax the VNET CPU for some segments. The number of interrupts is not large in this MSS. The ***(insert application aircraft)*** MSS has the segment Application wait until the VNET software is done with the request. This design is based on the requirement that Application data in messages must be immediately readable after a "Get" request and must be immediately writable after a "Put" request. The Application is blocked until the data transfers to or from the VNET. Application designers must assume that the very next instruction the Application executes after a Put request is a write to the buffer and that the very next instruction the Application executes after a Get request is a read from the buffer.

(This paragraph must be tailored to indicate the actual message interrupt strategy utilized in the application aircraft MSS. Alternative, acceptable designs exist. Since the largest number of requests a typical segment makes are Gets and since the segment typically calls the same Gets at the start of every frame 1, every frame 2, and so on, it made sense to provide a Get_List function. Get_List permits the VNET Interface to check over a whole list of messages, copy over the data and update all the return statuses in one call.)

3.2.4.1.4.2 VNET Start Up. The ***(insert application aircraft)*** MSS VNET, upon the application of power, performs a self test to verify functionality. Upon verification of successful functionality, the VNET attaches itself to the FDDI fiber optic ring per the FDDI standards. If the VNET should fail the test, it does not attach itself to the ring but instead provides a visual indication of a failure. The FDDI ring remains unbroken, and this FDDI node will continue to pass through traffic from other nodes.

(This paragraph must be tailored to indicate the actual startup capability provided in the application aircraft MSS. An optical bypass, for each FDDI connection, is strongly suggested. This will allow the ring to be operational even if a particular segment is not on-line for any reason.)

3.2.4.1.4.3 Token Control. The ***(insert application aircraft)*** MSS VNET utilizes a token rotation time of a maximum of ***(insert maximum time value)*** millisecond(s). The IOS segment can change the token rotation time as required.

(This paragraph must be tailored to indicate the actual token rotation utilized in the application aircraft MSS. The token rotation timer may be set by the standard FDDI contention or set by the IOS. However, with FDDI/XTP, the time cannot exceed 1.0 msec. The suggested time is 0.06 msec.)

3.2.4.1.4.4 VNET Performance Monitoring. The ***(insert application aircraft)*** MSS VNET supports runtime performance verification. Message delivery statistics are recorded in each segment as required to verify performance (i.e., message transmission time, message receipt time, content errors). During runtime, the VNET executes the VNET interface software, XTP, and any necessary background housekeeping tasks, such as management of module synchronization and communication with the FDDI board.

(This paragraph must be tailored to indicate the actual performance monitoring capability provided in the application aircraft MSS.)

3.2.4.1.4.5 Data Flow. The ***(insert application aircraft)*** MSS VNET utilizes the following data flow. As messages are received from the VNET interface, and processed by the application services layer, the segment application layer notifies the application services layer of the messages it is expecting to receive/send. The segment application layer processes received messages as they arrive. The segment application

layer has the capability to establish the importance of a given message and to interpret the message contents. The status of an application node is available to an application to make control decisions.

Messages are passed between the application services layer and the VNET interface layer in the appropriate message format. The VNET interface layer performs any necessary format translations for transmission through the VNET. The application services layer has the ability to request the status of the last operation in order to make decisions on important message transfers. Each segment's VNET interface layer has access to only to those messages used by that particular segment.

(These paragraphs must be tailored to indicate the actual data flow utilized in this application aircraft MSS.)

3.2.4.1.5 <u>VNET Design Characteristics</u>. The following paragraphs describe the general behavior and characteristics of the ***(insert application aircraft)*** MSS VNET design.

3.2.4.1.5.1 <u>Data Integrity</u>. The heart of VNET reliability is data integrity. The Application must be in complete control of the transfer of data to and from the VNET. The VNET Interface does not write or read the Application's data area except when requested by the Application.

There is always a possibility that the VNET Interface might be in the process of reading in a message at the very time that the Application asks for that message. Since the message is incomplete, the VNET Interface does not make the message available to a Get call until the whole message has been transferred in from the VNET.

(This paragraph must be tailored to discuss integrity in the application aircraft MSS. The paragraph implies that the Application must be able to call functions to command the VNET, with an assurance that accidental transfer will not occur. For example, the Application can execute a pair of equations like:

> ***A := X + 1;***
> ***B := X + 2;***

with the assurance that B will be one greater than A (that is, the value of X cannot change between the two equations). The reverse operation is similar. For example:

> ***X := 1;***
> ***Put(X);***
> ***X := 2;***

The second assignment (X := 2) will not cause the value of X to be overwritten in the process of being sent to the VNET.)

3.2.4.1.5.2 Error Handling. Any medium for data flow carries a possibility of error, for which the VNET design must account. Regardless of the medium, under no circumstances will the ***(insert application aircraft)*** MSS VNET Interface present bad or incomplete data to the Application. The ***(insert Application aircraft)*** MSS VNET transfer protocol is the Xpress Transfer Protocol (XTP). XTP meets the requirement for error detection and automatic retransmission of data at a very low cost in computational overhead.

(This paragraph must be tailored to identify the error handling capabilities for the Application aircraft MSS communication protocol. The tailoring may accomplish this goal through reference to other published material. The VNET Interface must be able to tell the Application about two special data conditions: (a) no data ever, and (b) no new data.)

3.2.4.1.5.3 Speed. The Application generates all message-sends (i.e., puts) no later than half way through a given frame. The ***(insert application aircraft)*** MSS VNET insures that valid messages arrive no later than the start of the next frame. This design ensures that the Applications have an independent execution order.

(This paragraph must be tailored to reflect the actual implementation, based on spare time requirements, for this Application aircraft MSS. The amount of time allotted for message transmission will vary from program to program. For example, in a simulator with a 50% spare frame requirement, it is known that no Application can ask the VNET Interface to send a message later than halfway through the frame. It is also known that the message must be at its destination segment no later than the start of the next frame. This translates into a half-frame requirement for message transmission. Other programs may impose stricter requirements. A design requirement like this prevents one Application from depending on the execution order of another Application.

Quickness of response to requests from the Application is very closely tied to hardware configurations and to program requirements. For example, it may be desirable in some hardware configurations for the Application to interrupt the VNET Interface process and to wait until that process has responded before the Application continues. In other configurations, it may be satisfactory for the VNET Interface to en queue the request and permit the Application to continue on at once. If the Application asks to receive a message, the VNET must never make the Application wait until the message has arrived.)

3.2.4.1.5.4 Legal Senders. A requirement of a MSS is that a given message may be sent by only one segment. If multiple segments were allowed to send a message, there is a severe chance that data would be overwritten or out of sync. The ***(insert application aircraft)*** MSS VNET design accounts for this problem. The VNET Interface must be able to notify the Application when it is attempting to send a message for

which it is not a legal sender, and there must be a method of identifying legal senders of messages to the VNET. There are two implications of this design: (a) an Application must be able to identify itself to the VNET, and (b) the VNET must be able to access a list of messages and their legal senders. This is done before the Application is allowed to send any messages.

(This paragraph should not requiring tailoring. Although multiple senders is not likely, this is more than simply a robust design. The most probable source for multiple senders arises from the presence of different software baselines in different segments.)

3.2.4.1.5.5 Legal Receivers. The ***(insert application aircraft)*** MSS VNET interface design also insists that the legal recipients of a message be identified beforehand. Segments are not allowed to "snoop" on messages unless they are marked as receivers of the message. This permits the VNET to manage data transmission in an economical way.

3.2.4.1.5.6 Multi-Copied Messages. The VNET Interface must be able to store multiple copies of a message and present them to the Application. The VNET informs the Application of how many messages of the type that it has as a return value from the Get call. In addition, it labels them as to arrival order and age.

(This paragraph must be tailored to identify the design utilized in the application aircraft MSS. Multi-copied messages are a potentially real problem. Imagine that a hypothetical automated IOS page responds to an instructor selection by setting Malfunction A and then clearing Malfunction B. Each of the malfunction messages is of the same message type. If the VNET Interface for a segment receiving the malfunction messages had space in its buffers for only one copy of a message, it could potentially receive both messages, but would overwrite the message that arrived first with the message that arrived second. As a result, the segment Application code would not know to set Malfunction B.)

3.2.4.1.6 VNET Design Capabilities. The ***(insert application aircraft)*** MSS VNET possess several capabilities in addition to inter-segment message transfer during runtime.

3.2.4.1.6.1 VNET Testing. The ***(insert application aircraft)*** MSS VNET can be tested without a real segment application present. The VNET includes two special segment applications both with no functionality: one to send a message and one to receive it.

(This paragraph must be tailored to indicate the exact level of VNET testing capability present in the application aircraft MSS.)

3.2.4.1.6.2 Message Counting. The ***(insert application aircraft)*** MSS VNET keeps a running count of the number of messages of each

type that each segment sends and receives. Examining these counts can confirm that a given message was sent a certain number of times by a given segment, and that the message was received the same number of times by each of the segments which was entitled to receive it. This is implemented in Application Services code in order to correct for any errors in the VNET Interface. If a message arrives at Application Services, it can be assumed to be available to the Application. A second message count is used to debug the Application Services. The count is kept by the Application Executive when it calls Put and Get and is compared with the count kept by Application Services. The two should always match if Application Services is functioning properly.

(This paragraph must be tailored to indicate the actual message counting approach utilized in the application aircraft MSS. These counts have been shown to be useful in integration. Very often a segment's misbehavior can be traced to a message not being sent or the message not being received. They can also serve to identify a bad network interface.)

3.2.4.1.6.3 Message Capture. The ***(insert application aircraft)*** MSS VNET provides a facility for capturing a specified message transfer.

(This paragraph must be tailored to indicate the actual message capture capability supported by the application aircraft MSS. The protocols surrounding messages very often provide checksums to avoid data errors between segments. Nevertheless, it is often useful to be able to capture an individual message at the Application Services level to verify visibly that the message was received exactly as it was transmitted. It is very desirable to be able to turn message capture on and off at will.)

3.2.4.1.6.4 Message Latency. The ***(insert application aircraft)*** MSS VNET provides a capability for measuring message latency. The VNET includes a special segment Application that sends only one message at a fixed rate, in response to an internal clock interrupt. A second special segment Application requests that the VNET Interface interrupt the Application when that message arrived. The delta in the send and receive time is a measure of the delay a message experiences in transversing the VNET.

The maximum allowable latency for the VNET is ***(insert maximum time value)*** microsecond(s). This is the maximum time that a VNET can take to receive a message for transmission from the host (application), pack the message and pass it to FDDI. It is also the maximum time that a VNET can take to receive a packed message, place it in memory, and notify the host that it is present.

(This paragraph must be tailored to indicate the actual latency measurement capability that the application aircraft MSS provides. Also, the discussion of message latency in

the application aircraft MSS should be tailored. Using FDDI/XTP, the maximum latency necessary to eliminate bottlenecks is 550 microseconds. It is suggested that this latency be reduced when a silicon (integrated circuit) version of XTP becomes commercially available to reduce the VNET bottleneck.)

3.2.4.1.6.5 Execution Timing. The ***(insert application aircraft)*** MSS VNET provides a capability to measure execution time. Gross overruns (in which a frame or frames overrun so bad as to never catch up) can be detected via the message counting capability. Minor overruns (in which a frame overruns but is made up in the next frame) can be detected via the VNET's timing functions: Start_Timing and Stop_Timing. These utilities use a circular array of 100 integers in which the elapsed time between the two calls was stored. This utility allows timing capture to be turned on and off at critical points. The utility also allows various sections of code to be timed. For example, it is possible to zero in on code which is taking a long time to execute, preferably without having to recompile.

(This paragraph must be tailored to indicate the actual execution timing capability that the application aircraft MSS provides.)

3.2.4.1.6.6 File Transfer Protocol. The ***(insert application aircraft)*** MSS VNET supports the non-real-time transfer of files between segments. If support for file transfers during real-time operation is provided, such support must use a lower priority than that used to transfer VNET message traffic, and not interfere with or slow down such traffic.

(This paragraph must be tailored to indicate the actual file transfer capability provided in the application aircraft MSS. If real time file transfer support is not required for the specific application, then the references may be removed.)

3.2.4.2 Message Data Representation. The following paragraphs present the design structure of interface messages.

(This paragraph and its sub-paragraphs must be tailored to specify the appropriate machine level details for this application aircraft MSS. Modular simulators may be implemented using several different kinds of computational hardware on the same system.

By isolating these factors in the VNET, no segment Application program need be aware of the kind of hardware it is running on, except for the topmost Executive. Even there, this machine data could be stored in a data file or perhaps be derived from operating system calls. Other implications of assigning Endian functions to the VNET are that the VNET may run on any sort of CPU architecture, and that the Application and VNET need not run on the same kind of CPU. The VNET knows the format requirements of the net and it knows the format requirements of its client Application CPUs; it merely needs to satisfy them.)

3.2.4.2.1 Identification Numbers. All interface messages include a unique 32 bit integer identification number. Identification numbers be used in the address field of an XTP information segment. The Global Message Types package includes an enumeration type which consists of all message names, and is built from the interface specification. The VNET utilizes this type to ensure the correct configuration of messages between segments in the system. A particular segment's application layer only has access to the inputs and outputs which are specifically identified as applicable to the segment in the ***(insert application aircraft)*** MSS IDD. A unique 4-byte (32 bit) integer is assigned to each message name. This integer is the identification number. The same integer values be assigned to the enumerated types, regardless of compiler implementation, given that the master message package is identical in each segment.

(This paragraph must be tailored to indicate the method of message identification utilized in the application aircraft's VNET implementation.)

3.2.4.2.2 Network Byte Order. The network byte order is as specified in MIL-STD-1777, Internet Protocol Specification.

The eight bits of each byte are transmitted on the media in the order that would be read in a left to right fashion, going from low to high memory locations. Where the left most bit is the Most Significant Bit (MSB) and the right most bit is the Least Significant Bit (LSB).

Bytes are also transmitted from left to right, from high order to low order. Whenever a multi-octet field represents a numerical quantity, the left most bit of the whole field is the most significant bit. When a multi-octet quantity is transmitted, the most significant octet is transmitted first. This is known in the computer industry as Big-Endian format.

The byte number is the offset in memory from the lowest byte or the base byte. The base byte is the address used when addressing the long word or half word as a single entity. Therefore, Byte 3 is a higher memory location address than Byte 0.

(This paragraph must be tailored to account for different CPUs utilized within the same implementation. Some CPUs store multi-byte data in memory with the most significant byte in the lower memory address and the least significant byte in the higher memory address (so called Big-Endian architectures). Other machines do the opposite: the most significant byte is stored in the higher memory address (so called Little-Endian architectures).

Data on the VNET, when implemented as a serial network, is defined in MIL-STD-1777, the Internet Protocol Specification, Appendix A. This is a Big-Endian approach, though this does not matter very much since the only requirement is that some sort of defined process exist in the VNET to transfer the data from memory to the bus, and from the bus to memory; this process could operate as easily in Little-Endian order as in Big-Endian order.

What should also be obvious is that the same process for copying serial data to memory cannot operate on both Big-Endian and Little-Endian CPUs and still preserve the meaning of the data. Accordingly, data must be converted somewhere, by some process. Who is responsible for this?

The answer is given in ISO-7498, the Open Systems Interconnect standard. It is the responsibility of the Presentation Layer of the ISO/OSI to perform data conversions of this type. This is handled automatically in the VNET. The segment Application software simply needs to inform the VNET whether it is a Big-Endian or Little-Endian machine.)

3.2.4.2.3 Basic Data Types. The following data types are used on the VNET:

a. short integer — 8 bit byte

b. integer — 16 bit double byte or half word

c. long integer — 32 bit quad byte or (long) word

d. float — Single and double precision IEEE Floating Point (Spec No 754-1985)

The first element of double and single precision enumerated types on the network is represented by number 0 in the type declaration. Basic data types are defined in package Base_Types in Appendix A of this document.

(This paragraph must be tailored to account for differences in Ada compilers and how they represent enumerations. It is probably true that every Ada compiler gives a value of zero for the first item in an enumeration, one for the second, and so on. Nevertheless, it makes sense not to trust the compiler to do so when a small expenditure of effort in writing representation clauses for the enumeration types can ensure that the same values will be used across the system.

Also of concern are the floating point number representation. The floating point standard for the VNET is IEEE-754. Not every CPU uses this format, though newer CPUs adhere to this standard. When an older floating point format (IBM 360, Gould, etc.) is required by a segment, the VNET must convert between this format and the IEEE format.)

3.2.4.2.4 Complex Data Structures. Data structures are placed on the communication media in the order they are declared in the type definition. This order is defined as

left-to-right and then top-to-bottom. Sub-structures comply with the same interpretation.

For example:

```
type Data_Structure_II is record

     Latitude: Float;

     Longitude: Float;

end record;


type Data_Structure is record

     Altitude: Float;

     Heading: Float;

     Position: Data_Structure_II;

end record;
```

An object of type "Data_Structure" would be sent in the following order: Altitude, Heading, Latitude and Longitude.

(This paragraph must be tailored to account for the interface message objects documented in Appendix A of this IDD. If different Ada compilers are used on any of these segments, or if the data will be transferred to a system which has a different Ada compiler (e.g., from the IOS to the IOS console), then it is necessary to write representation specifications for those interface message objects. The reason for this is that Ada compilers treat records as abstract data objects. Compilers are free to add padding to records in order to give proper word alignment to data fields within records.

Ada compilers for different CPUs, or even different versions of the same Ada compiler for the same CPU, cannot be expected to lay out fields in data records the same way. The purpose of representation specifications is to force the compiler to lay out the data in the way described in the specification.)

4. NOTES

4.1 Acronyms and Abbreviations

ANSI	American National Standards Institute
CASE	Computer Aided Software Engineering
CSCI	Computer Software Configuration Item
ENV	Environment Segment
EW	Electronic Warfare Segment
FC	Flight Controls Segment
FD	Flight Dynamics Segment
FDDI	Fiber Distributed Data Interface
FS	Flight Station Segment
HWCI	Hardware Configuration Item
IDD	Interface Design Document
IOS	Instructor Operator Station Segment
IRS	Interface Requirements Specification
ISO/OSI	International Standards Organization for Open System Integration
MSS	Modular Simulator System
NAV	Navigation/Communication Segment
PHC	Physical Cues Segment
PRO	Propulsion Segment
PEI	Protocol Engines Inc.
RDR	Radar Segment
VIS	Visual Segment
VLSI	Very Large Scale Integration
VNET	Virtual Network
WPN	Weapon Segment
XTP	Xpress Transfer Protocol

(This paragraph must be tailored to reflect acronyms and abbreviations in the application aircraft MSS.)

4.2 Glossary

APPLICATION. The actual hardware and software that implements a particular segment's functionality and performance requirements (e.g., Flight Dynamics).

APPLICATION SERVICES. The interface between the Application and the VNET.

MODEL. Simulations of systems and entities and executed within the various MSS segments

MODULE. One element of the MSS computational system on which resides the CSCIs that accomplish the required functionality of one or more segments.

SEGMENT. One of the top level partition elements in the MSS concept, to which functional capabilities are allocated.

VIRTUAL NETWORK. The means for transmission of information between MSS segments. This may be implemented as a physical or logical or heterogeneous network.

(This paragraph must be tailored to reflect terminology usage in the application aircraft MSS.)

ACTIVE PAGE RECORD													
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR
1											
2											
3											
4											
5											
6											
7											
8											
9											
10											
11											
12											
13											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											
25											
26											
27											
28											
29											
30											
31											
32											
33											
34											
35											
36											
37											
38											
39											
40											

ACTIVE PAGE RECORD											
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR
A-1						A-41					
A-2						A-42					
A-3						A-43					
A-4						A-44					
A-5						A-45					
A-6						A-46					
A-7						A-47					
A-8						A-48					
A-9						A-49					
A-10						A-50					
A-11						A-51					
A-12						A-52					
A-13						A-53					
A-14						A-54					
A-15						A-55					
A-16						A-56					
A-17						A-57					
A-18						A-58					
A-19						A-59					
A-20						A-60					
A-21						A-61					
A-22						A-62					
A-23						A-63					
A-24						A-64					
A-25						A-65					
A-26						A-66					
A-27						A-67					
A-28						A-68					
A-29						A-69					
A-30						A-70					
A-31						A-71					
A-32						A-72					
A-33						A-73					
A-34						A-74					
A-35						A-75					
A-36						A-76					
A-37						A-77					
A-38						A-78					
A-39						A-79					
A-40						A-80					

ACTIVE PAGE RECORD											
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR
A-81						A-121					
A-82						A-122					
A-83						A-123					
A-84						A-124					
A-85						A-125					
A-86						A-126					
A-87						A-127					
A-88						A-128					
A-89						A-129					
A-90						A-130					
A-91						A-131					
A-92						A-132					
A-93						A-133					
A-94						A-134					
A-95						A-135					
A-96						A-136					
A-97						A-137					
A-98						A-138					
A-99						A-139					
A-100						A-140					
A-101						A-141					
A-102						A-142					
A-103						A-143					
A-104						A-144					
A-105						A-145					
A-106						A-146					
A-107						A-147					
A-108						A-148					
A-109						A-149					
A-110						A-150					
A-111						A-151					
A-112						A-152					
A-113						A-153					
A-114						A-154					
A-115						A-155					
A-116						A-156					
A-117						A-157					
A-118						A-158					
A-119						A-159					
A-120						A-160					

ACTIVE PAGE RECORD											
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO	REV LTR
A-161						A-201					
A-162						A-202					
A-163						A-203					
A-164						A-204					
A-165						A-205					
A-166						A-206					
A-167						A-207					
A-168						A-208					
A-169						A-209					
A-170						A-210					
A-171						A-211					
A-172						A-212					
A-173						A-213					
A-174						A-214					
A-175						A-215					
A-176						A-216					
A-177						A-217					
A-178						A-218					
A-179						A-219					
A-180						A-220					
A-181						A-221					
A-182						A-222					
A-183						A-223					
A-184						A-224					
A-185						A-225					
A-186						A-226					
A-187						A-227					
A-188						A-228					
A-189						A-229					
A-190						A-230					
A-191						A-231					
A-192						A-232					
A-193						A-233					
A-194						A-234					
A-195						A-235					
A-196						A-236					
A-197						A-237					
A-198						A-238					
A-199						A-239					
A-200						A-240					

ACTIVE PAGE RECORD													
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR
A-241						A-281					
A-242						A-282					
A-243						A-283					
A-244						A-284					
A-245						A-285					
A-246						A-286					
A-247						A-287					
A-248						A-288					
A-249						A-289					
A-250						A-290					
A-251						A-291					
A-252						A-292					
A-253						A-293					
A-254						A-294					
A-255						A-295					
A-256						A-296					
A-257						A-297					
A-258						A-298					
A-259						A-299					
A-260						A-300					
A-261						A-301					
A-262						A-302					
A-263						A-303					
A-264						A-304					
A-265						A-305					
A-266						A-306					
A-267						A-307					
A-268						A-308					
A-269						A-309					
A-270						A-310					
A-271						A-311					
A-272						A-312					
A-273						A-313					
A-274						A-314					
A-275						A-315					
A-276						A-316					
A-277						A-317					
A-278						A-318					
A-279						A-319					
A-280						A-320					

ACTIVE PAGE RECORD											
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR
A-321						A-361					
A-322						A-362					
A-323						A-363					
A-324						A-364					
A-325						A-365					
A-326						A-366					
A-327						A-367					
A-328						A-368					
A-329						A-369					
A-330						A-370					
A-331						A-371					
A-332						A-372					
A-333						A-373					
A-334						A-374					
A-335						A-375					
A-336						A-376					
A-337						A-377					
A-338						A-378					
A-339						A-379					
A-340						A-380					
A-341						A-381					
A-342						A-382					
A-343						A-383					
A-344						A-384					
A-345						A-385					
A-346						A-386					
A-347						A-387					
A-348						A-388					
A-349						A-389					
A-350						A-390					
A-351						A-391					
A-352						A-392					
A-353						A-393					
A-354						A-394					
A-355						A-395					
A-356						A-396					
A-357						A-397					
A-358						A-398					
A-359						A-399					
A-360						A-400					

ACTIVE PAGE RECORD											
		ADDED PAGES						ADDED PAGES			
PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR	PAGE NO.	REV LTR
A-401											
A-402											
A-403											
A-404											
A-405											
A-406											
A-407											
A-408											
A-409											
A-410											
A-411											
A-412											
A-413											
A-414											
A-415											
A-416											
A-417											
A-418											
A-419											
A-420											

ACTIVE PAGE RECORD												
PAGE NO.	REV LTR	ADDED PAGES				PAGE NO.	REV LTR	ADDED PAGES				
		PAGE NO.	REV LTR	PAGE NO.	REV LTR			PAGE NO.	REV LTR	PAGE NO.	REV LTR	
B-1												
B-2												
B-3												
B-4												
B-5												
B-6												
B-7												
B-8												

APPENDIX A

Inter-segment Interfaces

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

TABLE OF CONTENTS

```
-----------------------------------------------------------------------
-- %Z% Unit Name:       Global_Message_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  19 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package contains definitions of types which are contained in
--  messages sent by more than one segment, with the exception of:
--     (a) Base_Types (e.g., integer, float)
--     (b) Engineering_Units (e.g., Mach number, Degrees, Radians)
--     (c) Moving_Model_Types (e.g., threats, companion vehicles)
--     (d) Control_Types (e.g., responses to IOS messages)
--     (e) Service_Function_Types (e.g., Occulting, Spatial Relations)
--
-- Adaptation:
--   Section 10.1.1 will require modification to meet the requirements
--   of the particular aircraft being simulated and the type of
--   simulator used.  Determine the values of each of the enumeration
--   types listed below; comment out or delete those which do not
--   apply to this simulator/aircraft; and add other values as
--   required.  Section 10.1.2 will not commonly require any
--   modification.
--
with Base_Types;
with Engineering_Units;
with Control_Types;
--
package Global_Message_Types is
--
--***************************************************************
--*                                                             *
--/ 10.1   Global Message Types
--*                                                             *
```

```
--************************************************************

Bytes : constant := 8; -- improve readability of rep specs

--****************************************
--*
--/ 10.1.1   Aircraft/Simulator-Specific Global Types
--*
--****************************************

-- This section of the package will contain the data types
-- which will be specific to the simulated aircraft. It may
-- or may not require modification when moved from project
-- to project.  The types in Section 10.1.2 (Reusable Global
-- Types) are not intended to change from program to program,
-- while types in this current section will vary.


--**************************
--*
--/ 10.1.1.1   Aircraft Configuration Type Definitions
--*
--**************************

type Crew_Station is
   (Pilot);
for Crew_Station'size use 8;

type Aircraft_Electrical_Bus is (
   Emergency_AC_Bus_No_1,
   Emergency_AC_Bus_No_2,
   Essential_AC_Bus_No_2,
   Nonessential_AC_Bus_No_1,
   Nacelle_Essential_AC_Bus,
   Nacelle_Nonessential_AC_Bus,
   Nonessential_AC_Bus_No_2,
   Radar_AC_Bus,
   Overcurrent_Protection_Panel_1,
   Emergency_DC_Bus_No_1,
   Emergency_DC_Bus_No_2,
   Essential_DC_Bus_No_2,
```

```
   Nonessential_DC_Bus_No_1,
   Nacelle_Noness_DC_Bus,
   Battery_Bus_No_1,
   Battery_Bus_No_2);
Aircraft_Electrical_Bus_Size : constant := 8;
for Aircraft_Electrical_Bus'size use Aircraft_Electrical_Bus_Size;

Number_Of_Aircraft_Electrical_Busses : constant :=
   Aircraft_Electrical_Bus'pos (Aircraft_Electrical_Bus'last) -
   Aircraft_Electrical_Bus'pos (Aircraft_Electrical_Bus'first) + 1;

type Aircraft_Electrical_Generator is (
   Main_Generator,
   Standby_Generator,
   EPU_Generator);
for Aircraft_Electrical_Generator'size use 8;

--NOTE: Aircraft Hydraulic Components must be ordered by segment
type Aircraft_Hydraulic_Component is (
   -- ENV
   -- EW
   EW_TBD,
   -- FC
   Left_Flaperon,
   Left_Horizontal_Stabilizer,
   Right_Flaperon,
   Right_Horizontal_Stabilizer,
   Rudder,
   Nosewheel,
   Left_Landing_Gear,
   Nose_Landing_Gear,
   Right_Landing_Gear,
   Left_Wheel_Brake,
   Right_Wheel_Brake,
   Left_Leading_Edge_Flap,
   Right_Leading_Edge_Flap,
   Right_Speedbrake,
   Left_Speedbrake,
   Air_Refuel_Door,
   Left_Landing_Gear_Door,
   Nose_Landing_Gear_Door,
```

```
   Right_Landing_Gear_Door,
   Gun_Purge_Door,
   -- FD
   Drag_Chute,
   -- FS
   -- IOS
   -- NAV
   -- PHC
   -- PRO
   EPU,
   -- RDR
   RDR_TBD,
   -- VIS
   VIS_TBD,
   -- WPN
   WPN_TBD);
for Aircraft_Hydraulic_Component'size use 8;

subtype Electronic_Warfare_Hydraulic_Component is
   Aircraft_Hydraulic_Component range
   EW_TBD..EW_TBD;

subtype Flight_Controls_Hydraulic_Component is
   Aircraft_Hydraulic_Component range
   Left_Flaperon..Gun_Purge_Door;

subtype Flight_Dynamics_Hydraulic_Component is
   Aircraft_Hydraulic_Component
   range Drag_Chute..Drag_Chute;

subtype Propulsion_Hydraulic_Component is
   Aircraft_Hydraulic_Component
   range EPU..EPU;

subtype Radar_Hydraulic_Component is
   Aircraft_Hydraulic_Component range
   RDR_TBD..RDR_TBD;

subtype Visual_System_Hydraulic_Component is
   Aircraft_Hydraulic_Component range
   VIS_TBD..VIS_TBD;
```

```
subtype Weapons_Hydraulic_Component is
   Aircraft_Hydraulic_Component range
   WPN_TBD..WPN_TBD;

type Aircraft_Hydraulic_Pump is (
   System_A_Pump,
   System_B_Pump,
   EPU_Pump);
for Aircraft_Hydraulic_Pump'size use 8;

type Aircraft_Hydraulic_System is (
   System_A,
   System_B);
for Aircraft_Hydraulic_System'size use 8;

type Aircraft_Hydraulic_Reservoir is (
   System_A_Reservior,
   System_B_Reservior);
Aircraft_Hydraulic_Reservoir_Size : constant := 32;
for Aircraft_Hydraulic_Reservoir'size use
   Aircraft_Hydraulic_Reservoir_Size;

type Aircraft_Pneumatic_Component is (
   Left_Landing_Gear,
   Right_Landing_Gear,
   Nose_Landing_Gear);
for Aircraft_Pneumatic_Component'size use 8;

subtype Flight_Controls_Pneumatic_Component is
   Aircraft_Pneumatic_Component range
   Left_Landing_Gear..
   Nose_Landing_Gear;

type Aircraft_Oxygen_System is (
   Main_System,
   Emergency_System);
for Aircraft_Oxygen_System'size use 8;

type Aircraft_Doors_And_Hatches is (
   Air_Refuel,
```

```
   Left_Landing_Gear,
   Nose_Landing_Gear,
   Right_Landing_Gear,
   Gun_Purge);
for Aircraft_Doors_And_Hatches'size use 8;

type Aircraft_Fuel_Tank is (
   Left_Wing,
   Right_Wing,
   Centerline_External,
   Left_External,
   Right_External,
   F_1,
   F_2,
   Aft_Reservior,
   Fwd_Reservior,
   A_1);
for Aircraft_Fuel_Tank'size use 8;

type Primary_Control_Surface is (
   Left_Flaperon,
   Left_Horizontal_Stabilizer,
   Right_Flaperon,
   Right_Horizontal_Stabilizer,
   Left_Horizontal_Tail,
   Right_Horizontal_Tail,
   Rudder,
   Nosewheel);
for Primary_Control_Surface'size use 8;

type Secondary_Control_Surface is (
   Left_Leading_Edge_Flap,
   Left_Speedbrake,
   Right_Leading_Edge_Flap,
   Right_Speedbrake,
   Nosewheel);
for Secondary_Control_Surface'size use 8;

type Aircraft_Landing_Gear is (
   Left,
   Nose,
```

```
   Right);
for Aircraft_Landing_Gear'size use 8;

type Aircraft_Wheel is (
   Left,
   Nose,
   Right);
for Aircraft_Wheel'size use 8;

type Aircraft_Surface_Tab is (To_Be_Determined);
for Aircraft_Surface_Tab'size use 8;

type Cockpit_Control_Device is (
   Pitch_Stick,
   Roll_Stick,
   Left_Pedal,
   Right_Pedal);
for Cockpit_Control_Device'size use 8;

type Aircraft_Unique_Control_Device is (Ram_Air_Turbine);
for Aircraft_Unique_Control_Device'size use 8;

type Aircraft_Engine is (
   Engine);
Aircraft_Engine_Size : constant := 32;
for Aircraft_Engine'size use Aircraft_Engine_Size;

type Aircraft_APU is (
   EPU);
for Aircraft_APU'size use 8;

type Aircraft_Throttle_Lever is (
   Throttle);
for Aircraft_Throttle_Lever'size use 8;

type IFF_Mode is (
   Mode_1,
   Mode_2,
   Mode_3A,
   Mode_4,
   Mode_C);
```

```
for IFF_Mode'size use 16;

type Master_Mode is (
   Nav,
   A_A,
   A_G);
for Master_Mode'size use 8;

type SMS_Submode is (
   Off,
   Stby,
   INV,
   AAM,
   A_G,
   E_J,
   S_J);
for SMS_Submode'size use 8;

type A_G_Weapon_Delivery_Mode is (
   MAN,
   LADD,
   DTOS,
   CCRP,
   CCIP);
for A_G_Weapon_Delivery_Mode'size use 8;

type Release_Option is (
   Sgl,
   Pair);
Release_Option_Size : constant := 8;
for Release_Option'size use Release_Option_Size;

type Arming_Option is (
   Nose,
   Tail,
   Nstl);
Arming_Option_Size : constant := 8;
for Arming_Option'size use Arming_Option_Size;

type Fuze_Arming is (
   Impact,
```

```
   Altitude,
   Arming_Time,
   Rockeye);
Fuze_Arming_Size : constant := 8;
for Fuze_Arming'size use Fuze_Arming_Size;

type Weapon_Profile is (
   Profl,
   Prof2);
Weapon_Profile_Size : constant := 8;
for Weapon_Profile'size use Weapon_Profile_Size;

type Weapon_Status is (
   Rdy,
   Sim,
   Rel,
   Mal,
   Saf,
   None);
Weapon_Status_Size : constant := 8;
for Weapon_Status'size use Weapon_Status_Size;

type Jettison_Status is (
   Hot,
   None);      -- RGC 01JUN90 --
for Jettison_Status'size use 8;

type Jettison_Type is (
   Selective,
   Emergency,
   None);
Jettison_Type_Size : constant := 8;
for Jettison_Type'size use Jettison_Type_Size;

type Stores_Station is (
   Station_1,
   Station_2,
   Station_3,
   Station_3_Rack_1,
   Station_3_Rack_2,
   Station_3_Rack_3,
```

```
      Station_4,
      Station_4_Rack_1,
      Station_4_Rack_2,
      Station_4_Rack_3,
      Station_5,
      Station_5_Rack_1,
      Station_5_Rack_2,
      Station_5_Rack_3,
      Station_6,
      Station_6_Rack_1,
      Station_6_Rack_2,
      Station_6_Rack_3,
      Station_7,
      Station_7_Rack_1,
      Station_7_Rack_2,
      Station_7_Rack_3,
      Station_8,
      Station_9);
   Stores_Station_Size : constant := 8;
   for Stores_Station'size use Stores_Station_Size;

   type Station_Weapon_Load is (
      Empty,                          Fuel_Tank_300_Gallon,
      Fuel_Tank_370_Gallon,           Fuel_Tank_600_Gallon,
      Mxu_648_Travel_Pod,             An_Alq_119_Ecm_Pod,
      An_Alq_131_Ecm_Pod,             Mau_12,
      Mer_10n_10,                     Ter_9a,
      Suu_20,                         Lau_3,
      Lau_68,                         Lau_88_A,
      Lau_88_Aa,                      Lau_117_A,
      Lau_5003,                       Launcher,
      Aim_9_Launcher,                 Lau_129_Wrl_W,
      Lau_129_Mod,                    Mk_36,
      M_129e2,                        Mk_106,
      Gun_Ammo,                       Blu_27b_B,
      Blu_27b_Bf,                     Blu_52,
      Mc_1,                           Blu_33,
      Bdu_33,                         Matra_250,
      Bsu_49,                         Bsu_49b,
      Bsu_491,                        Bsu_49bl,
      Mk_82,                          Mk_82a,
```

```
   Mk_82s,                          Mk_82sba,
   Mk_82sld,                        Mk_82s_Ba_Ld,
   Bsu_50,                          Bsu_50b,
   Bsu_50l,                         Bsu_50bl,
   Mk_84,                           Mk_84_Ba,
   Mk_20_Ad,                        Mk_20_Ba,
   Bl_755_Ad,                       Bl_755_Ba,
   Cbu_52_Ad,                       Cbu_52_Ba,
   Cbu_58_Ad,                       Cbu_58_Ba,
   Cbu_71_Ad,                       Cbu_71_Ba,
   Cbu_87_Ad,                       Cbu_87_Ba,
   Cbu_89_Ad,                       Cbu_89_Ba,
   Nbdu_33,                         Nmk_106,
   Mk_5,                            M156,
   M_151,                           Ra_79,
   Cm_151,                          Mk_61,
   M151,                            Cm151,
   B57_P,
   B57_Modl,                        B57_Pf,
   B57_F,                           B61_G,
   B61_J,                           B61_G2,
   B61_Gl,                          B612_J2,
   B61_Jl,                          Gbu_10,
   Gbu_10c_B,                       Gbu_12,
   Agm_65a,                         Agm_65b,
   Agm_65d,                         Aim_9j,
   Aim_9np,                         Aim_9lm,
   Aim_120a,                        Itv,
   Target_37u_33,                   Tow_Cable_37u_33);
Station_Weapon_Load_Size : constant := 8;
for Station_Weapon_Load'size use Station_Weapon_Load_Size;

type Station_Status is (
   Empty,
   Release,
   Load,
   Hung,
   Jettison);
for Station_Status'size use 32;
```

```
--**************************
--*
--/ 10.1.1.2   Constants for this simulator
--*
--**************************

-- Constants which only apply to messages sent by one segment should
-- be declared in the output interface types package for that segment.

Maximum_Number_Of_Waypoints        : constant := 10;
Max_IFF_Code_Length                : constant := 6;


--****************************************
--*
--/ 10.1.2   Aircraft/Simulator Reusable Global Types
--*
--****************************************

-- This section contains the global types that will be used
-- from project to project and to the extent possible,
-- usable across several segments.  The data types in this
-- section should not be changed.

--**************************
--*
--/ 10.1.2.1   Aircraft Configuration Type Definitions
--*
--**************************

-- EW, FC, FD, NAV, PRO, RDR, VIS and WPN each send FS
-- electrical bus loads for their equipment

type Aircraft_Electrical_Bus_Load_Array is
   array (Aircraft_Electrical_Bus) of Engineering_Units.Amperes;
Aircraft_Electrical_Bus_Load_Array_Size : constant :=
   Number_Of_Aircraft_Electrical_Busses * 4 * Bytes;
for Aircraft_Electrical_Bus_Load_Array'size use
   Aircraft_Electrical_Bus_Load_Array_Size;
```

```
-- FD sends fuel quantity in Weight and Balance
-- FS sends fuel quantity in Fuel Management System

Number_Of_Fuel_Tanks : constant :=
  Aircraft_Fuel_Tank'pos (Aircraft_Fuel_Tank'last) -
  Aircraft_Fuel_Tank'pos (Aircraft_Fuel_Tank'first) + 1;
type Fuel_Tank_Quantity_Array is
   array (Aircraft_Fuel_Tank) of Engineering_Units.Pounds;
Fuel_Tank_Quantity_Array_Size : constant :=
   Number_Of_Fuel_Tanks  * 4 * Bytes;
for Fuel_Tank_Quantity_Array'size use
   Fuel_Tank_Quantity_Array_Size;


-- FD and PRO use Vibration_Characteristics

type Vibration_Characteristics is
 record
   Frequency : Engineering_Units.Hertz;
   Amplitude : Engineering_Units.Inches;
end record;
for Vibration_Characteristics use
 record
   Frequency at 0 range 0..31;
   Amplitude at 4 range 0..31;
end record;
Vibration_Characteristics_Size : constant := 8 * Bytes;
for Vibration_Characteristics'size use
   Vibration_Characteristics_Size;


-- FS uses Fluid_Characteristics for hydraulic fluid
-- PRO uses Fluid_Characteristics for engine oil

type Fluid_Characteristics is
 record
   Quantity    : Engineering_Units.Gallons;
   Pressure    : Engineering_Units.PSI;
   Temperature : Engineering_Units.Degrees_C;
end record;
for Fluid_Characteristics use
```

```
 record
   Quantity at    0 range 0..31;
   Pressure at    4 range 0..31;
   Temperature at 8 range 0..31;
end record;
Fluid_Characteristics_Size : constant := 12 * Bytes;
for Fluid_Characteristics'size use
   Fluid_Characteristics_Size;


--**************************
--*
--/ 10.1.2.2   Common/Shared Segment Type Definitions
--*
--**************************



--**********
--*
--/ 10.1.2.2.1   IFF Definition
--*
--**********
--
--   used by NAV, RDR, EW, FS, ENV

subtype IFF_Code is String(1..Max_IFF_Code_Length);
-- unitless code string

type IFF_Data is
 record
   Mode: IFF_Mode;
   Code: IFF_Code;
end record;
for IFF_Data use
 record
   Mode at 0 range  0..15;
   Code at 0 range 16..63;
end record;
IFF_Data_Size : constant := 8 * Bytes;
for IFF_Data'size use IFF_Data_Size;
```

```
--**********
--*
--/ 10.1.2.2.2   INS Waypoint Definition
--*
--**********

subtype INS_Waypoints is Base_Types.Unsigned_Integer_16 range
   1..Maximum_Number_Of_Waypoints;
INS_Waypoints_Size : constant := 16;

type Waypoint_Change is
 record
   New_Waypoint    : Engineering_Units.Earth_Position_Components;
   Waypoint_Number : INS_Waypoints;
end record;
for Waypoint_Change use
 record
   New_Waypoint    at  0 range 0..159;
   Waypoint_Number at 20 range 0..15;
end record;
for Waypoint_Change'size use 22 * Bytes;


--**********
--*
--/ 10.1.2.2.3   Weapon stations and stores
--*
--**********

type Weapon_Station_Loading is
 record
   Station      : Stores_Station;
   Weapon       : Station_Weapon_Load;
   Quantity     : Base_Types.Signed_Integer_16;
   Status       : Station_Status;
 end record;
for Weapon_Station_Loading use
 record
   Station      at 0 range 0..7;
   Weapon       at 0 range 8..15;
```

```
      Quantity      at 0 range 16..31;
      Status        at 4 range 0..31;
   end record;
  Weapon_Station_Loading_Size : constant := 8 * Bytes;
  for Weapon_Station_Loading'size use
     Weapon_Station_Loading_Size;

  Number_Of_Stores_Stations : constant :=
     Stores_Station'pos (Stores_Station'last) -
     Stores_Station'pos (Stores_Station'first) + 1;

  subtype Stores_Station_Count is Base_Types.Unsigned_Integer_32
     range 1..Number_Of_Stores_Stations;

  type Weapon_Station_Loading_Array is
       array (Stores_Station_Count) of Weapon_Station_Loading;
  for Weapon_Station_Loading_Array'size use 192 * Bytes;

  type Weapon_Station_Change is
   record
     Number_Of_Stations           : Stores_Station_Count;
     Stores                       : Weapon_Station_Loading_Array;
  end record;
  for Weapon_Station_Change use
   record
     Number_Of_Stations           at 0 range 0..31;
     Stores                       at 4 range 0..(192 * Bytes)-1;
  end record;
  for Weapon_Station_Change'size use 196 * Bytes;

  --**********
  --*
  --/ 10.1.3      Numeric Types
  --*
  --**********

  -- see Primitive_Types and Base_Types

  end Global_Message_Types;
```

```
--------------------------------------------------------------------
-- %Z% Unit Name:       Primitive_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Jim Richardson, Bob Crispen, et al.
-- %Z% Date of Origin:  19 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
--------------------------------------------------------------------
--
-- Purpose:
--  This package is used as a base for Base_Types
--
-- Use:
--  This package should not be directly WITHed.  Instead, WITH the
--  package Base_Types.
--
package Primitive_Types is
--
--*************************************************************
--*                                                           *
--/ 10.1.3.1  Primitive Numeric Types
--*                                                           *
--*************************************************************

   -- Float_64 range - 4.49423283715579E+307 .. 4.49423283715579E+307
   type Float_64       is digits 15;
   for Float_64'Size use 64;
   --
   -- Bits       : 64
   -- Digits     : 15
   -- Mantissa   : 51 BITs
   -- Epsilon    : 8.88178419700125E-16
   -- Emax       : 204
   -- Small      : 1.94469227433161E-62
   -- Large      : 2.57110087081438E+61
   -- Safe_Emax  : 1022
   -- Safe_Small : 1.11253692925360E-308
```

```
-- Safe_Large : 4.49423283715579E+307
--

-- Float_32 range  - 8.50706E+37 .. 8.50706E+37
type Float_32 is digits 6;
for Float_32'Size use 32;
--
-- BITs        : 32
-- Digits      : 6
-- Mantissa    : 21 BITs
-- Epsilon     : 9.53674E-07
-- Emax        : 84
-- Small       : 2.58494E-26
-- Large       : 1.93428E+25
-- Safe_Emax   : 126
-- Safe_Small  : 5.87747E-39
-- Safe_Large  : 8.50706E+37
--

-- NOTE:  Integer types can not exceed 32 bits
-- type Integer_64 can not be defined
-- type Unsigned_Integer_64 can not be defined

-- Integer_32 range is -16#80000000# .. 16#7FFFFFFF#
-- or -2_147_483_648 .. 2_147_483_647
type Integer_32 is range (-(2 ** 31)) .. ((2 ** 31) - 1);
for Integer_32'Size use 32;

-- NOTE:  Unsigned_Integer_32 should be 0 .. 16#FFFFFFFF# however
-- Ada-83 converts all literal integers it finds to Universal_Integer
-- before processing further. Any Integer over 16#7FFFFFFF# raises
-- Constraint_Error. Therefore, the actual range has been set to
-- 0 .. 16#7FFFFFFF# or 0 .. 2_147_483_647 and the size is 32 bits.
-- The top bit is not used.  One possible alternative is to define
-- Unsigned_Integer_32 based on a vendor-supplied 32-bit unsigned
-- integer type, but this is not done here because it would be
-- grossly non-portable.

type Unsigned_Integer_32  is range 0 .. ((2 ** 31) - 1);
for Unsigned_Integer_32'Size use 32;
```

```
   -- Integer_16 range is -16#8000# .. 16#7FFF#
   -- or -32768 .. 32767
   type Integer_16 is range (-(2 ** 15)) .. ((2 ** 15) - 1);
   for Integer_16'Size use 16;

   -- Unsigned_Integer_16 range is 0..16#FFFF#
   -- or 0 .. 65535
   type Unsigned_Integer_16  is range 0 .. 16#FFFF#;
   for Unsigned_Integer_16'Size use 16;

   -- Integer_8 range is -16#80# .. 16#7F#
   -- or -128 .. 127
   type Integer_8 is range (-(2 ** 7)) .. ((2 ** 7) - 1);
   for Integer_8'Size use 8;

   -- Unsigned_Integer_8 range is 0 .. 16#FF#
   -- or 0 .. 255
   type Unsigned_Integer_8  is range 0 .. 16#FF#;
   for Unsigned_Integer_8'Size use 8;

   -- Unsigned_Integer_1 range 0 .. 1
   --
   -- NOTE:  The Following Implemenation Would Not
   -- compile under some of the older Ada Compilers,
   -- therefore the enumeration version was adopted.
   -- (We don't need arithmetic operators for a single
   -- bit anyhow, just set and clear capability).
   --
   -- type Unsigned_Integer_1 is range 0 .. 1;
   -- for Unsigned_Integer_1'Size use 1;
   type Unsigned_Integer_1  is ('0', '1');
   for Unsigned_Integer_1 use
      ('0' => 0, '1' => 1);
   for Unsigned_Integer_1'size use 1;

end Primitive_Types;
```

```
---------------------------------------------------------------------
-- %Z% Unit Name:         Base_Types
-- %Z% Source Pathname:   %P%
-- %Z% Unit Type:         Package Spec (no body)
-- %Z% Unit ID:           (tbd)
-- %Z% Author:            Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:    19 August 1993
-- %Z% SCCS Filename:     %M%
-- %Z% Delta ID:          %I%
-- %Z% Delta Date:        %G%
-- %Z% Current Release:   %R%
---------------------------------------------------------------------
--
-- Purpose:
--  Use of types from package Standard should be avoided because
--  those types differ in size.  Use types declared in this package
--  instead.
--
-- Adaptation:
--  Do all machine-specific adaptation on the package Primitive_Types.
--
with Primitive_Types;
--
package Base_Types is
--
--**************************************************************
--*                                                            *
--/ 10.1.3.2    Base Numeric Types
--*                                                            *
--**************************************************************

   -- Float_64 range is - 4.49423283715579E+307 .. 4.49423283715579E+307
   type Float_64 is new Primitive_Types.Float_64;
   type Float_64_Access is access Base_Types.Float_64;

   -- Float_32 range is - 8.50706E+37 .. 8.50706E+37
   type Float_32 is new Primitive_Types.Float_32;
   type Float_32_Access is access Base_Types.Float_32;

   -- Signed_Integer_32 range -2_147_483_648 .. 2_147_483_647
   type Signed_Integer_32 is new Primitive_Types.Integer_32;
```

```
   type Signed_Integer_32_Access is access Base_Types.Signed_Integer_32;

   -- NOTE:  Unsigned_Integer_32 should be 0 .. 16#FFFFFFFF# however
   -- Ada-83 converts all literal integers it finds to Universal_Integer
   -- before processing further. Any Integer over 16#7FFFFFFF# raises
   -- Constraint_Error. Therefore, the actual range has been set to
   -- 0 .. 16#7FFFFFFF# or 0 .. 2_147_483_647 and the size is 32 bits.
   -- The top bit is not used.  One possible alternative is to define
   -- Unsigned_Integer_32 based on a vendor-supplied 32-bit unsigned
   -- integer type, but this is not done here because it would be
   -- grossly non-portable.

   -- Unsigned_Integer_32 range 0 .. 2_147_483_647
   type Unsigned_Integer_32 is new Primitive_Types.Unsigned_Integer_32;
   type Unsigned_Integer_32_Access is access
Base_Types.Unsigned_Integer_32;

   -- Natural_Integer_32 range 0.. 2_147_483_647
   subtype Natural_Integer_32 is
      Base_Types.Signed_Integer_32
      range 0 .. Base_Types. Signed_Integer_32'Last;

   type Natural_Integer_32_Access is access Base_Types.Natural_Integer_32;

   -- Positive_Integer_32 range 1.. 2_147_483_647
   subtype Positive_Integer_32 is
      Base_Types.Natural_Integer_32
      range 1 .. Base_Types. Natural_Integer_32'Last;

   type Positive_Integer_32_Access is access
Base_Types.Positive_Integer_32;

   -- Signed_Integer_16 range -32768 .. 32767
   type Signed_Integer_16 is new Primitive_Types.Integer_16;
   type Signed_Integer_16_Access is access Base_Types.Signed_Integer_16;

   -- Unsigned_Integer_16 range 0 .. 65535
   type Unsigned_Integer_16 is new Primitive_Types.Unsigned_Integer_16;
   type Unsigned_Integer_16_Access is access
Base_Types.Unsigned_Integer_16;

   -- Natural_Integer_16 range 0 .. 32767
```

```
subtype Natural_Integer_16 is
   Base_types.Signed_Integer_16
   range 0 .. Base_Types. Signed_Integer_16'Last;

type Natural_Integer_16_Access is access Base_Types.Natural_Integer_16;

-- Positive_Integer_16 range 1 .. 32767
subtype Positive_Integer_16 is
   Base_types.Natural_Integer_16
   range 1 .. Base_Types. Natural_Integer_16'Last;

type Positive_Integer_16_Access is access
Base_Types.Positive_Integer_16;

-- Signed_Integer_8 range -128 .. 127
type Signed_Integer_8 is new Primitive_Types.Integer_8;
type Signed_Integer_8_Access is access Base_Types.Signed_Integer_8;

-- Unsigned_Integer_8 range 0 .. 255
type Unsigned_Integer_8 is new Primitive_Types.Unsigned_Integer_8;
type Unsigned_Integer_8_Access is access Base_Types.Unsigned_Integer_8;

-- Natural_Integer_8 range 0 .. 127
subtype Natural_Integer_8 is
   Base_Types.Signed_Integer_8
   range 0 .. Base_Types. Signed_Integer_8'Last;

type Natural_Integer_8_Access is access Base_Types.Natural_Integer_8;

-- Positive_Integer_8 range 1 .. 127
subtype Positive_Integer_8 is
   Base_Types.Natural_Integer_8
   range 1 .. Base_Types. Natural_Integer_8'Last;

type Positive_Integer_8_Access is access Base_Types.Positive_Integer_8;

-- Unsigned_Integer_1 range 0 .. 1
type Unsigned_Integer_1 is new Primitive_Types.Unsigned_Integer_1;
type Unsigned_Integer_1_Access is access Base_Types.Unsigned_Integer_1;

-- Discrete state: Off, On
type Discrete_State is (Off, On);
```

```
   for Discrete_State use (Off => 0, On => 1);
   for Discrete_State'size use 8;

   -- Boolean: False, True
   --  Note: Only Standard.Boolean allows Ada to evaluate expressions
like
   --      if (Boolean_Variable) then...
   --  but if you declare Sim_Boolean as a subtype of Standard.Boolean,
   --  you won't be able to rep spec it (or rather, the rep specs for
   --  records containing Sim_Booleans will fail for machine
architectures
   --  that don't have 8-bit Booleans.  Bottom line: it's a tradeoff.
   type Sim_Boolean is (False, True);
   for Sim_Boolean use (False => 0, True => 1);
   for Sim_Boolean'size use 8;
   -- Alternative (see note above):
   -- subtype Sim_Boolean is Boolean;

end Base_Types;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Engineering_Units
-- %Z% Source Pathname:  %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   19 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release:  %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package supplies subtypes which are used throughout the
--  simulation.  Whenever a variable is declared inside a component,
--  consideration should be given to using one of the types below,
--  since (e.g.) a declaration of Hydraulic_Pressure of type PSI
--  gives much greater self-documentation than defining it as Float_32.
--
-- Adaptation:
--  The only changes forseen for this package are (a) changes in
--  precision (e.g., changing Latitude_Position from double-precision
--  to single-precision) and (b) changes having to do with the
mathematical
--  model or aircraft model employed (e.g., Weight in Kilograms rather
--  than Pounds).  Many times, no changes at all will be required in
--  this package.  Since changes in these types will affect more than
--  one segment, care should be taken to ensure consensus on the
--  engineering units being employed in this simulator.
--
with Base_Types;
--
package Engineering_Units is
--
--*************************************************************
--*                                                           *
--/ 10.1.4          Engineering Units
--*                                                           *
--*************************************************************
```

```
-- Declarations to make representation specs more readable
Bytes : constant := 8;

-- Use this constant only for supplying range limits for (e.g.)
-- radians.  This gives us a little slop so we don't run into
-- constraint errors.  Use Sim_Math.Pi for a better value.
Pi : constant := 3.1416;


subtype Pressure is Base_Types.Float_32;
--
subtype PSI is Pressure;
-- pounds per square inch
subtype Inches_Hg is Pressure;
-- inches of Mercury (barometric pressure)


subtype Weight is Base_Types.Float_32;
--
subtype Pounds is Weight;
-- lbs. of weight


subtype Slope is Base_Types.Float_32;
-- rise unit/run unit
subtype Inches_Per_Foot is Slope;
-- inches rise/foot run
subtype Feet_Per_Feet is Slope;
-- feet rise/feet run


subtype Temperature is Base_Types.Float_32;
--
subtype Degrees_C is Temperature;
-- degrees centigrade
subtype Degrees_F is Temperature;
-- degrees farenheit


subtype Linear_Velocity is Base_Types.Float_32;
-- linear unit/unit time
```

```
subtype Feet_Per_Sec is Linear_Velocity;
-- feet/sec
subtype Hertz is Linear_Velocity;
-- 1/sec
subtype MHz is Linear_Velocity;
-- 1_000_000/sec
subtype Knots is Linear_Velocity;
-- nautical miles/hour
subtype Ft_Per_Min is Linear_Velocity;
-- feet/min

subtype Angular_Velocity is Base_Types.Float_32;
-- angular unit/unit time
subtype Radians_Per_Sec is Angular_Velocity;
-- radians/sec
subtype RPM is Angular_Velocity;
-- revolutions/minute
subtype Double_Precision_Angular_Velocity is Base_Types.Float_64;
-- angular unit/unit time
subtype Double_Precision_Radians_Per_Sec is
Double_Precision_Angular_Velocity;
-- radians/sec

subtype Flow_Velocity is Base_Types.Float_32;
-- quantity unit/unit time
subtype Lbs_Per_Hour is Flow_Velocity;
-- lbs/hr
subtype Gal_Per_Min is Flow_Velocity;
-- gallons/min
subtype Ft3_Per_Min is Flow_Velocity;
-- cubic feet/min


subtype Torque is Base_Types.Float_32;
--
subtype Ft_Lbs is Torque;


subtype Linear_Acceleration is Base_Types.Float_32;
-- linear unit/unit time2
subtype Feet_Per_Sec2 is Linear_Acceleration;
-- feet/sec2
```

```
subtype Angular_Acceleration is Base_Types.Float_32;
-- angular unit/unit time2
subtype Radians_Per_Sec2 is Angular_Acceleration;
-- radians/sec2
subtype Double_Precision_Angular_Acceleration is Base_Types.Float_64;
-- angular unit/unit time2
subtype Double_Precision_Radians_Per_Sec2 is
   Double_Precision_Angular_Acceleration;
-- radians/sec2


subtype Length is Base_Types.Float_32;
-- linear units
subtype Feet is Length;
-- feet units
subtype Inches is Length;
-- inch units
subtype Nautical_Miles is Length;
-- nautical mile units
subtype Double_Precision_Length is Base_Types.Float_64;
-- linear units
subtype Double_Precision_Feet is Double_Precision_Length;
-- feet units
subtype Double_Precision_Inches is Double_Precision_Length;
-- inch units
subtype Double_Precision_Nautical_Miles is Double_Precision_Length;
-- nautical mile units


subtype Current is Base_Types.Float_32;
--
subtype Amperes is Current;
--electrical current unit


subtype Time is Base_Types.Float_32;
--
subtype Hours is Time range 0.0..24.0;
-- 24 hrs. per day
```

```
subtype Minutes is Time range 0.0..60.0;
-- 60 mins. per hour
subtype Seconds is Time range 0.0..60.0;
-- 60 secs. per min
subtype Time_In_Seconds is Time;


subtype Power_Unit is Base_Types.Float_32;
-- unit of power
subtype Watts is Power_Unit;
-- watt units
subtype kVA is Power_Unit;
-- kilo Volt Amperes units


subtype Decibel is Base_Types.Float_32;
-- power ratio unit


subtype Lumens is Base_Types.Float_32;
-- unit of brightness


subtype Volume is Base_Types.Float_32;
--
subtype Gallons is Volume;
--
subtype Cubic_Feet is Volume;


subtype Percent is Base_Types.Float_32 range 0.0..100.0;


subtype Normalized is Base_Types.Float_32 range 0.0..1.0;
--
subtype Signed_Normalized is Base_Types.Float_32
                             range -1.0..1.0;


subtype Angular_Unit is Base_Types.Float_32;
--
```

```
subtype Degrees is Angular_Unit range 0.0..360.0;
--
subtype Signed_Degrees is Angular_Unit range -180.0..180.0;
--
subtype Radians is Angular_Unit range 0.0..2.0 * Pi;
--
subtype Signed_Radians is Angular_Unit range -Pi..Pi;
--
subtype Double_Precision_Angular_Unit is Base_Types.Float_64;
--
subtype Double_Precision_Radians is Double_Precision_Angular_Unit
   range 0.0..2.0 * Pi;
--
subtype Double_Precision_Degrees is Double_Precision_Angular_Unit
   range 0.0..360.0;
--
subtype Double_Precision_Signed_Degrees is Double_Precision_Angular_Unit
   range -180.0..180.0;
--
subtype Double_Precision_Signed_Radians is Double_Precision_Angular_Unit
   range -Pi..Pi;

subtype Mach_Range is Base_Types.Float_32 range 0.0..10.0;
-- Mach number

subtype Zero_To_Ten is Base_Types.Signed_Integer_32 range 0..10;
-- Gives a scalar value between zero and ten, used (e.g.) for
-- Thunderstorm intensity, moving model complexity, etc.


subtype Gravity is Base_Types.Float_32 range -10.0..10.0;
-- vertical acceleration/gravitational acceleration


type Linear_Position_Components is
 record
   Longitudinal_Position : Feet;
   Lateral_Position      : Feet;
   Vertical_Position     : Feet;
 end record;
for Linear_Position_Components use
```

```
 record
   Longitudinal_Position at 0 range 0..31;
   Lateral_Position      at 4 range 0..31;
   Vertical_Position     at 8 range 0..31;
 end record;
Linear_Position_Components_Size : constant := 12 * Bytes;
for Linear_Position_Components'size use
   Linear_Position_Components_Size;

type Angular_Position_Components is
 record
   Roll_Angle  : Signed_Radians;
   Pitch_Angle : Signed_Radians;
   Yaw_Angle   : Radians;
 end record;
for Angular_Position_Components use
 record
   Roll_Angle  at 0 range 0..31;
   Pitch_Angle at 4 range 0..31;
   Yaw_Angle   at 8 range 0..31;
 end record;
Angular_Position_Components_Size : constant := 12 * Bytes;
for Angular_Position_Components'size use
   Angular_Position_Components_Size;

type Earth_Position_Components is
 record
   Latitude_Position  : Double_Precision_Signed_Radians;
   Longitude_Position : Double_Precision_Signed_Radians;
   Altitude_Position  : Feet;
 end record;
for Earth_Position_Components use
 record
   Latitude_Position   at  0 range 0..63;
   Longitude_Position  at  8 range 0..63;
   Altitude_Position   at 16 range 0..31;
 end record;
Earth_Position_Components_Size : constant := 20 * Bytes;
for Earth_Position_Components'size use
   Earth_Position_Components_Size;
```

```
type Linear_Velocity_Components is
 record
   Longitudinal_Velocity  : Feet_Per_Sec;
   Lateral_Velocity       : Feet_Per_Sec;
   Vertical_Velocity      : Feet_Per_Sec;
 end record;
for Linear_Velocity_Components use
 record
   Longitudinal_Velocity at 0 range 0..31;
   Lateral_Velocity      at 4 range 0..31;
   Vertical_Velocity     at 8 range 0..31;
 end record;
Linear_Velocity_Components_Size : constant := 12 * Bytes;
for Linear_Velocity_Components'size use
   Linear_Velocity_Components_Size;

type Angular_Velocity_Components is
 record
   Roll_Velocity  : Radians_Per_Sec;
   Pitch_Velocity : Radians_Per_Sec;
   Yaw_Velocity   : Radians_Per_Sec;
 end record;
for Angular_Velocity_Components use
 record
   Roll_Velocity  at 0 range 0..31;
   Pitch_Velocity at 4 range 0..31;
   Yaw_Velocity   at 8 range 0..31;
 end record;
Angular_Velocity_Components_Size : constant := 12 * Bytes;
for Angular_Velocity_Components'size use
   Angular_Velocity_Components_Size;

type Earth_Velocity_Components is
 record
   North_South_Velocity : Feet_Per_Sec;
   East_West_Velocity   : Feet_Per_Sec;
   Altitude_Velocity    : Feet_Per_Sec;
 end record;
for Earth_Velocity_Components use
 record
   North_South_Velocity at 0 range 0..31;
```

```
    East_West_Velocity   at 4 range 0..31;
    Altitude_Velocity    at 8 range 0..31;
  end record;
Earth_Velocity_Components_Size : constant := 12 * Bytes;
for Earth_Velocity_Components'size use
    Earth_Velocity_Components_Size;

type Linear_Acceleration_Components is
  record
    Longitudinal_Acceleration : Feet_Per_Sec2;
    Lateral_Acceleration      : Feet_Per_Sec2;
    Vertical_Acceleration     : Feet_Per_Sec2;
  end record;
for Linear_Acceleration_Components use
  record
    Longitudinal_Acceleration at 0 range 0..31;
    Lateral_Acceleration      at 4 range 0..31;
    Vertical_Acceleration     at 8 range 0..31;
  end record;
Linear_Acceleration_Components_Size : constant := 12 * Bytes;
for Linear_Acceleration_Components'size use
    Linear_Acceleration_Components_Size;

type Angular_Acceleration_Components is
  record
    Roll_Acceleration  : Radians_Per_Sec2;
    Pitch_Acceleration : Radians_Per_Sec2;
    Yaw_Acceleration   : Radians_Per_Sec2;
  end record;
for Angular_Acceleration_Components use
  record
    Roll_Acceleration  at 0 range 0..31;
    Pitch_Acceleration at 4 range 0..31;
    Yaw_Acceleration   at 8 range 0..31;
  end record;
Angular_Acceleration_Components_Size : constant := 12 * Bytes;
for Angular_Acceleration_Components'size use
    Angular_Acceleration_Components_Size;

type Earth_Acceleration_Components is
  record
```

```
    North_South_Acceleration : Feet_Per_Sec2;
    East_West_Acceleration   : Feet_Per_Sec2;
    Altitude_Acceleration    : Feet_Per_Sec2;
  end record;
 for Earth_Acceleration_Components use
  record
    North_South_Acceleration at 0 range 0..31;
    East_West_Acceleration   at 4 range 0..31;
    Altitude_Acceleration    at 8 range 0..31;
  end record;
 Earth_Acceleration_Components_Size : constant := 12 * Bytes;
 for Earth_Acceleration_Components'size use
    Earth_Acceleration_Components_Size;

 type Lat_Long_Location is
  record
    Latitude  : Signed_Radians;
    Longitude : Signed_Radians;
  end record;
 for Lat_Long_Location use
  record
    Latitude  at 0 range 0..31;
    Longitude at 4 range 0..31;
  end record;
 Lat_Long_Location_Size : constant := 8 * Bytes;
 for Lat_Long_Location'size use Lat_Long_Location_Size;

 type Polar_Direction is
  record
    Azimuth   : Signed_Radians;
    Elevation : Signed_Radians;
  end record;
 for Polar_Direction use
  record
    Azimuth   at 0 range 0..31;
    Elevation at 4 range 0..31;
  end record;
 Polar_Direction_Size : constant := 8 * Bytes;
 for Polar_Direction'size use Polar_Direction_Size;

 end Engineering_Units;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Control_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   19 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--   This package contains primarily types which are used in messages
--   sent from other segments to the IOS or Control segment in
--   response to IOS commands.  Whenever a type is required both in
--   the message and in its response, it is declared here.
--
-- Adaptation:
--
with Base_Types;
with Engineering_Units;
--
package Control_Types is
--
-----------------------------------------------------------------------
--
--/ 10.1.5          Control Types
--
-----------------------------------------------------------------------

Bytes : constant := 8;

--***************************
--*
--/ 10.1.5.1   Aircraft/Simulator Specific Control Types
--*
--***************************

Maximum_Iteration_Rate     : constant := 50;
```

```
Maximum_Frame_Number      : constant := 16;

-- Names of subsystems: used in the subsystem timing on-line diagnostic
-- messages, and in the scheduling tables.  The subsystems correspond
-- one-for-one with the Mod Sim functions.
--
-- Two things will be required to adapt this table:
--  (a) Determine in which segments (if any) the four service functions
are
--      to be performed, and comment them out of the other segments.
--  (b) Delete functions which are not to be performed in this simulator.
-- In unusual cases it may be necessary to add functions to segments.
type Subsystems is (
   No_Subsystem,
        -- ENV
   MSE_Interaction,                          -- 10.24.3.1
   Atmosphere,                               -- 10.24.3.2
   Ownship_Weapons_Damage_Assessment,        -- 10.24.3.3
   Threat_Weapon_Dynamics,                   -- 10.24.3.4
   External_Entity,                          -- 10.24.3.5
   External_Entity_Chaff_and_Flares,         -- 10.24.3.6
   Database_Management,                      -- 10.24.3.7
   Threat_Environment_Database,              -- 10.24.3.8
   Navigation_Environment,                   -- 10.24.3.9
   Collision_Detection,                      -- 10.24.3.10
   Radar_Database,                           -- 10.24.3.11 (Service
Function)
   Visual_Database,                          -- 10.24.3.12 (Service
Function)
   Spatial_Relations,                        -- 10.24.3.13 (Service
Function)
   Occulting,                                -- 10.24.3.14 (Service
Function)
   Environment_Support,                      -- 10.24.3.15
        -- EW
   Ownship_Chaff_And_Flares,                 -- 10.16.3.1
   Dedicated_Displays,                       -- 10.16.3.2
   Ownship_ECM,                              -- 10.16.3.3
   Pods_And_Controls,                        -- 10.16.3.4
   Radar_Warning_Receiver,                   -- 10.16.3.5
   Threat_Detection,                         -- 10.16.3.6
   Electronic_Warfare_Support,               -- 10.16.3.7
```

```
      -- FC
 Primary_Controls,                             -- 10.4.3.1
 Misc_Control_Devices,                         -- 10.4.3.2
 Trim,                                         -- 10.4.3.3
 Toe_Brakes_And_Anti_Skid,                     -- 10.4.3.4
 AFCS,                                         -- 10.4.3.5
 Hinge_Moments,                                -- 10.4.3.6
 Flight_Controls_Support,                      -- 10.4.3.7
      -- FD
 Equations_Of_Motion,                          -- 10.6.3.1
 Weight_And_Balance,                           -- 10.6.3.2
 Forces_And_Moments,                           -- 10.6.3.3
 Envelope_Violation,                           -- 10.6.3.4
 Flight_Dynamics_Support,                      -- 10.6.3.5
      -- FS
 Electrical_System,                            -- 10.2.3.1
 Hydraulic_System,                             -- 10.2.3.2
 Fuel_Management_System,                       -- 10.2.3.3
 Pneumatic_System,                             -- 10.2.3.4
 Autochecklist_System,                         -- 10.2.3.5
 Oxygen_System,                                -- 10.2.3.6
 Crew_Station_Interface,                       -- 10.2.3.7
 Flight_Station_Support,                       -- 10.2.3.8
      -- IOS
 Simulator_Control,                            -- 10.22.3.1
 Ownship_Status_And_Control,                   -- 10.22.3.2
 Ownship_Malfunction,                          -- 10.22.3.3
 Ownship_Controls_Disagreement,                -- 10.22.3.4
 Nav_Comm_Status_And_Control,                  -- 10.22.3.5
 Natural_Environment_Status_And_Control,       -- 10.22.3.6
 Tactical_Environment_Status_And_Control,      -- 10.22.3.7
 Performance_Monitoring_And_Measurement,       -- 10.22.3.8
 IOS_Support,                                  -- 10.22.3.9
      -- NAV
 AHRS,                                         -- 10.10.3.1
 INS,                                          -- 10.10.3.2
 Radar_Altimeter,                              -- 10.10.3.3
 Radio_Navigation_Aids,                        -- 10.10.3.4
 Communications,                               -- 10.10.3.5
 IFF,                                          -- 10.10.3.6
 Star_Tracker,                                 -- 10.10.3.7
```

```
    Doppler_Radar,                          -- 10.10.3.8
    ADS,                                    -- 10.10.3.9
    Navigation_Communication_Support,       -- 10.10.3.10
         -- PHC
    Environmental_Sound,                    -- 10.18.3.1
    Anti_G_Suit,                            -- 10.18.3.2
    G_Seat,                                 -- 10.18.3.3
    Motion_Geometry,                        -- 10.18.3.4
    Motion_Cue,                             -- 10.18.3.5
    Motion_Base,                            -- 10.18.3.6
    Vibration_And_Buffet,                   -- 10.18.3.7
    Physical_Cues_Support,                  -- 10.18.3.8
         -- PRO
    Engine_Inlet,                           -- 10.8.3.1
    Core_Engine,                            -- 10.8.3.2
    Thrust_Generation,                      -- 10.8.3.3
    Starting,                               -- 10.8.3.4
    Engine_Bleed_Air,                       -- 10.8.3.5
    Transmission,                           -- 10.8.3.6
    Auxiliary_Power_Unit,                   -- 10.8.3.7
    Engine_Fuel,                            -- 10.8.3.8
    Engine_Exhaust,                         -- 10.8.3.9
    Engine_Oil,                             -- 10.8.3.10
    Propulsion_Support,                     -- 10.8.3.11
         -- RDR
    Radar_Processor,                        -- 10.14.3.1
    Radar_Image_Generation,                 -- 10.14.3.2
    Airborne_Interrogate_Sensor,            -- 10.14.3.3
    TF_TA_Radar_Guidance,                   -- 10.14.3.4
    Mission_Computer_Interface,             -- 10.14.3.5
    Radar_Aircraft_Systems_Interface,       -- 10.14.3.6
    Crew_Station_Hardware_Panel_Interface,  -- 10.14.3.7
    -- Radar_Database,                          -- 10.14.3.8  (Service
Function)
    -- Visual_Database,                         -- 10.14.3.9  (Service
Function)
    -- Spatial_Relations,                       -- 10.14.3.10 (Service
Function)
    -- Occulting,                               -- 10.14.3.11 (Service
Function)
    Radar_Support,                          -- 10.14.3.12
         -- VIS
```

```
   Image_Generation,                         -- 10.20.3.1
   Moving_Model,                             -- 10.20.3.2
   Visual_Scene_Environment,                 -- 10.20.3.3
   Lighting,                                 -- 10.20.3.4
   Mission_Computer_DP_Interface,            -- 10.20.3.5
   Visual_Crew_Station_Interface,            -- 10.20.3.6
   Visual_Aircraft_Systems_Interface,        -- 10.20.3.7
   Visual_Display_Systems,                   -- 10.20.3.8
   -- Radar_Database,                           -- 10.20.3.9  (Service
Function)
   -- Visual_Database,                          -- 10.20.3.10 (Service
Function)
   -- Spatial_Relations,                        -- 10.20.3.11 (Service
Function)
   -- Occulting,                                -- 10.20.3.12 (Service
Function)
   Visual_Support,                           -- 10.20.3.13
        -- WPN
   Ownship_Fire_Control,                     -- 10.12.3.1
   Ownship_Weapon_Dynamics,                  -- 10.12.3.2
   Ownship_Weapons_Stores,                   -- 10.12.3.3
   Target_Designation,                       -- 10.12.3.4
   Threat_Weapons_Damage_Assessment,         -- 10.12.3.5
   Weapons_Support                           -- 10.12.3.6
);
Subsystems_Size : constant := 16;
for Subsystems'Size use Subsystems_Size;


-- These subtype declarations should match the appropriate section
-- of the list above.
subtype Environment_Subsystems is Subsystems range
   MSE_Interaction..Environment_Support;

subtype Electronic_Warfare_Subsystems is Subsystems range
   Ownship_Chaff_And_Flares..Electronic_Warfare_Support;

subtype Flight_Controls_Subsystems is Subsystems range
   Primary_Controls..Flight_Controls_Support;

subtype Flight_Dynamics_Subsystems is Subsystems range
   Equations_Of_Motion..Flight_Dynamics_Support;
```

```
subtype Flight_Station_Subsystems is Subsystems range
   Electrical_System..Flight_Station_Support;

subtype IOS_Subsystems is Subsystems range
   Simulator_Control..IOS_Support;

subtype Navigation_Communication_Subsystems is Subsystems range
   AHRS..Navigation_Communication_Support;

subtype Physical_Cues_Subsystems is Subsystems range
   Environmental_Sound..Physical_Cues_Support;

subtype Propulsion_Subsystems is Subsystems range
   Engine_Inlet..Propulsion_Support;

subtype Radar_Subsystems is Subsystems range
   Radar_Processor..Radar_Support;

subtype Visual_Subsystems is Subsystems range
   Image_Generation..Visual_Support;

subtype Weapons_Subsystems is Subsystems range
   Ownship_Fire_Control..Weapons_Support;

-- Names of components: used in the component timing on-line diagnostic
-- messages, and in the scheduling tables.  Adapt by adding functions
-- to the appropriate segments.
type Components is (
   No_Component,
   -- ENV
   MSE_Interaction,
   Atmosphere,
   Weather,
   Ownship_Height_Above_Terrain,
   Moving_Model_Height_Above_Terrain,
   Threat_Weapon_Dynamics,
   External_Entity,
   External_Entity_Chaff_and_Flares,
   Database_Management,
   Threat_Environment_Database,
```

```
    Navaids,
    Airports,
    Terrain_Database,
    Collision_Detection,
    Environment_Support,
    -- EW
    Electronic_Warfare_Support,
    -- FC
    Flight_Controls_Support,
    -- FD
    Flight_Dynamics_Support,
    -- FS
    Flight_Station_Support,
    -- IOS
    IOS_Support,
    -- NAV
    ILS,                                     -- 10.10.3.4.1
    TACAN,                                   -- 10.10.3.4.2
    MLS,                                     -- 10.10.3.4.3
    ADF,                                     -- 10.10.3.4.4
    GPS,                                     -- 10.10.3.4.5
    VOR,                                     -- 10.10.3.4.6
    LORAN,                                   -- 10.10.3.4.7
    OMEGA,                                   -- 10.10.3.4.8
    SKE,                                     -- 10.10.3.4.9
    UHF_VHF_HF_Intercom,                     -- 10.10.3.5.1
    SATCOM,                                  -- 10.10.3.5.2
    JTIDS,                                   -- 10.10.3.5.3
    Navigation_Communication_Support,
    -- PHC
    Physical_Cues_Support,
    -- PRO
    Propulsion_Support,
    -- RDR
    Radar_Support,
    -- VIS
    Visual_Support,
    -- WPN
    Weapons_Support
);
Components_Size : constant := 16;
```

```
for Components'size use Components_Size;

-- These subtype declarations should match the appropriate section
-- of the list above.
subtype Environment_Components is Components range
   MSE_Interaction..Environment_Support;

subtype Electronic_Warfare_Components is Components range
   Electronic_Warfare_Support..Electronic_Warfare_Support;

subtype Flight_Controls_Components is Components range
   Flight_Controls_Support..Flight_Controls_Support;

subtype Flight_Dynamics_Components is Components range
   Flight_Dynamics_Support..Flight_Dynamics_Support;

subtype Flight_Station_Components is Components range
   Flight_Station_Support..Flight_Station_Support;

subtype IOS_Components is Components range
   IOS_Support..IOS_Support;

subtype Navigation_Communication_Components is Components range
   ILS..Navigation_Communication_Support;

subtype Physical_Cues_Components is Components range
   Physical_Cues_Support..Physical_Cues_Support;

subtype Propulsion_Components is Components range
   Propulsion_Support..Propulsion_Support;

subtype Radar_Components is Components range
   Radar_Support..Radar_Support;

subtype Visual_Components is Components range
   Visual_Support..Visual_Support;

subtype Weapons_Components is Components range
   Weapons_Support..Weapons_Support;
```

```
--***************************
--*
--/ 10.1.5.2   Aircraft/Simulator Reusable Control Types
--*
--***************************
subtype Frame_Number is Base_Types.Signed_Integer_32;
subtype Simulation_Frames is Frame_Number range 1..Maximum_Frame_Number;

-- A name for each segment and segment tester: used in the segment
-- timing on-line diagnostic request, and in several of the executive
-- and VNET packages.  Under normal circumstances, these will not
-- be modified, even if one or more segments are absent.
type Segment_And_Tester_Names is (
   No_Segment,
   IOS,
   Navigation_Communication,
   Flight_Controls,
   Flight_Dynamics,
   Flight_Station,
   Electronic_Warfare,
   Weapons,
   Propulsion,
   Radar,
   Visual,
   Physical_Cues,
   Environment,
   Module_Executive,
   Environment_Tester,
   Physical_Cues_Tester,
   Visual_Tester,
   Radar_Tester,
   Propulsion_Tester,
   Weapons_Tester,
   Electronic_Warfare_Tester,
   Flight_Station_Tester,
   Flight_Dynamics_Tester,
   Flight_Controls_Tester,
   Navigation_Communication_Tester,
   IOS_Tester);

Segment_And_Tester_Names_Size : constant := 16;
```

```
for Segment_And_Tester_Names'size use
   Segment_And_Tester_Names_Size;

-- These bit flags are extraordinarily important in the VNET, and
-- should not be changed under any circumstances.
for Segment_And_Tester_Names use (
   No_Segment                        => 2#0000000000000000#,
   IOS                               => 2#0000000000000001#,
   Navigation_Communication          => 2#0000000000000010#,
   Flight_Controls                   => 2#0000000000000100#,
   Flight_Dynamics                   => 2#0000000000001000#,
   Flight_Station                    => 2#0000000000010000#,
   Electronic_Warfare                => 2#0000000000100000#,
   Weapons                           => 2#0000000001000000#,
   Propulsion                        => 2#0000000010000000#,
   Radar                             => 2#0000000100000000#,
   Visual                            => 2#0000001000000000#,
   Physical_Cues                     => 2#0000010000000000#,
   Environment                       => 2#0000100000000000#,
   Module_Executive                  => 2#0001000000000000#,
   Environment_Tester                => 2#0111011111111111#,
   Physical_Cues_Tester              => 2#0111101111111111#,
   Visual_Tester                     => 2#0111110111111111#,
   Radar_Tester                      => 2#0111111011111111#,
   Propulsion_Tester                 => 2#0111111101111111#,
   Weapons_Tester                    => 2#0111111110111111#,
   Electronic_Warfare_Tester         => 2#0111111111011111#,
   Flight_Station_Tester             => 2#0111111111101111#,
   Flight_Dynamics_Tester            => 2#0111111111110111#,
   Flight_Controls_Tester            => 2#0111111111111011#,
   Navigation_Communication_Tester   => 2#0111111111111101#,
   IOS_Tester                        => 2#0111111111111110#);

subtype Segment_Names is Segment_And_Tester_Names range
   IOS..Environment;

-- Simulation State Change response
type Simulation_State_Responses is (
   Mission_Generation_Active,
   Training_Active,
   Shutdown,
```

```
   Local_Diagnostic_Active,
   Reset_Active,
   Remote_Controlled_Diagnostic_Active);
for Simulation_State_Responses'size use 16;


-- Training Mode Change response
type Training_Mode_Responses is (
   Initialized,
   Aligned_To_Approach,
   Aligned_To_Departure,
   Running,
   Totally_Frozen,
   Training_Terminated);
for Training_Mode_Responses'size use 16;


-- Performance Test
type Performance_Tests is (
   To_Be_Determined);
Performance_Tests_Size : constant := 8;
for Performance_Tests'size use Performance_Tests_Size;


-- Execution Timing requests and responses
type Structural_Element_To_Time is (
   No_Timing,
   Segment_Timing,
   Subsystem_Timing,
   Component_Timing);
Structural_Element_To_Time_Size : constant := 16;
for Structural_Element_To_Time'size use
   Structural_Element_To_Time_Size;


-- Based on a Unix struct: used to return the value of interval timers.
type Timeval is
 record
   tv_sec  : Base_Types.Signed_Integer_32; -- seconds
   tv_usec : Base_Types.Signed_Integer_32; -- microseconds
end record;
```

```
for Timeval use
 record
   tv_sec  at 0 range 0..31;
   tv_usec at 4 range 0..31;
end record;
for Timeval'size use 64;


-- Legal alignment points
type Alignments is (
   Approach,
   Departure);


-- Diagnostics
type Test_Results is (
   Running,
   Passed,
   Failed);
for Test_Results'size use 8;

type Task_Responses is (
   Initialized,
   Executing,
   On_Hold,
   Resumed,
   Aborted,
   Completed);
for Task_Responses'size use 8;

type Off_Line_Diagnostics is (
   To_Be_Determined);
Off_Line_Diagnostics_Size : constant := 8;
for Off_Line_Diagnostics'size use
   Off_Line_Diagnostics_Size;

type On_Line_Diagnostics is (
   To_Be_Determined);
On_Line_Diagnostics_Size : constant := 8;
for On_Line_Diagnostics'size use
   On_Line_Diagnostics_Size;
```

```
type On_Line_Diagnostic_Status is (
   Critical_Failure,
   Non_Critical_Failure,
   No_Failure,
   Not_Running);
for On_Line_Diagnostic_Status'size use 8;

type On_Line_Diagnostic_Status_Array is
   array (On_Line_Diagnostics) of
   On_Line_Diagnostic_Status;
for On_Line_Diagnostic_Status_Array'size use 1 * Bytes;

type Remote_Controlled_Diagnostics is (
   To_Be_Determined);
Remote_Controlled_Diagnostics_Size : constant := 8;
for Remote_Controlled_Diagnostics'size use
   Remote_Controlled_Diagnostics_Size;

-- Scoring
type Scoring_Attributes is (
   Weapons_Delivery,
   Refueling,
   Instrument_Landing,
   Glideslope_Deviation,
   Localizer_Deviation,
   Fuel_Efficientcy);
for Scoring_Attributes'size use 8;

subtype Scores is Base_Types.Signed_Integer_32;

-- Malfunctions
type Malfunction is
   (To_Be_Determined);
   --NAV,FD,FC,FS,PRO,EW,WPN,RDR

type Malfunction_Kind is (
   Switchable,
   Value_Driven);
```

```
--**************************
--*
--/ 10.1.5.3   Response Message Types
--*
--**************************

type Segment_Simulation_State_Response is
 record
   Responding_Segment : Segment_Names;
   Control_Response   : Simulation_State_Responses;
 end record;
for Segment_Simulation_State_Response use
 record
   Responding_Segment at 0 range 0..(2 * Bytes)-1;
   Control_Response   at 2 range 0..(2 * Bytes)-1;
 end record;
for Segment_Simulation_State_Response'size use 4 * Bytes;


type Segment_Training_Mode_Response is
 record
   Responding_Segment : Segment_Names;
   Control_Response   : Training_Mode_Responses;
 end record;
for Segment_Training_Mode_Response use
 record
   Responding_Segment at 0 range 0..(2 * Bytes)-1;
   Control_Response   at 2 range 0..(2 * Bytes)-1;
 end record;
for Segment_Training_Mode_Response'size use 4 * Bytes;


type Performance_Test_Response is
 record
   Responding_Segment : Segment_Names;
   Task_Response      : Task_Responses;
   Performance_Test   : Performance_Tests;
   Test_Result        : Test_Results;
 end record;
for Performance_Test_Response use
 record
```

```
    Responding_Segment at 0 range 0..(2 * Bytes)-1;
    Task_Response      at 2 range 0..(1 * Bytes)-1;
    Performance_Test   at 3 range 0..(1 * Bytes)-1;
    Test_Result        at 4 range 0..(1 * Bytes)-1;
  end record;
 for Performance_Test_Response'size use 5 * Bytes;


 type Remote_Controlled_Diagnostic_Response is
  record
    Responding_Segment           : Segment_Names;
    Task_Response                : Task_Responses;
    Remote_Controlled_Diagnostic : Remote_Controlled_Diagnostics;
    Test_Result                  : Test_Results;
  end record;
 for Remote_Controlled_Diagnostic_Response use
  record
    Responding_Segment           at 0 range 0..(2 * Bytes)-1;
    Task_Response                at 2 range 0..(1 * Bytes)-1;
    Remote_Controlled_Diagnostic at 3 range 0..(1 * Bytes)-1;
    Test_Result                  at 4 range 0..(1 * Bytes)-1;
  end record;
 for Remote_Controlled_Diagnostic_Response'size use 5 * Bytes;


 type Off_Line_Diagnostic_Response is
  record
    Responding_Segment  : Segment_Names;
    Task_Response       : Task_Responses;
    Off_Line_Diagnostic : Off_Line_Diagnostics;
    Test_Result         : Test_Results;
  end record;
 for Off_Line_Diagnostic_Response use
  record
    Responding_Segment  at 0 range 0..(2 * Bytes)-1;
    Task_Response       at 2 range 0..(1 * Bytes)-1;
    Off_Line_Diagnostic at 3 range 0..(1 * Bytes)-1;
    Test_Result         at 4 range 0..(1 * Bytes)-1;
  end record;
 for Off_Line_Diagnostic_Response'size use 5 * Bytes;
```

```
type On_Line_Diagnostic_Response is
 record
   Responding_Segment        : Segment_Names;
   On_Line_Diagnostic_Result : On_Line_Diagnostic_Status_Array;
 end record;
for On_Line_Diagnostic_Response use
 record
   Responding_Segment        at 0 range 0..(2 * Bytes)-1;
   On_Line_Diagnostic_Result at 2 range 0..(1 * Bytes)-1;
 end record;
for On_Line_Diagnostic_Response'size use 3 * Bytes;


type Timing_Response is
 record
   Segment   : Segment_Names;
   Subsystem : Subsystems;
   Component : Components;
   Request   : Structural_Element_To_Time;
   Time      : Timeval;
 end record;
for Timing_Response use
 record
   Segment   at 0 range 0..(2 * Bytes)-1;
   Subsystem at 2 range 0..(2 * Bytes)-1;
   Component at 4 range 0..(2 * Bytes)-1;
   Request   at 6 range 0..(2 * Bytes)-1;
   Time      at 8 range 0..(8 * Bytes)-1;
 end record,
for Timing_Response'size use 16 * Bytes;


type Scoring_Response is
 record
   Score             : Scores;
   Scoring_Attribute : Scoring_Attributes;
 end record;
for Scoring_Response use
 record
   Score             at 0 range 0..(4 * Bytes)-1;
```

```
      Scoring_Attribute at 4 range 0..(1 * Bytes)-1;
    end record;
   for Scoring_Response'size use 5 * Bytes;


   type Malfunction_Demand (
      Kind : Malfunction_Kind := Switchable) is
    record
      Name : Malfunction;
      State : Base_Types.Discrete_State;
      case Kind is
           When Switchable =>
                null;
           When Value_Driven =>
                Value : Base_Types.Float_32;
      end case;
    end record;
   for Malfunction_Demand use
    record
      Kind  at 0 range 0..(1 * Bytes)-1;
      Name  at 1 range 0..(1 * Bytes)-1;
      State at 2 range 0..(1 * Bytes)-1;
      Value at 4 range 0..(4 * Bytes)-1;
    end record;
   for Malfunction_Demand'size use 8 * Bytes;


   end Control_Types;
```

```
---------------------------------------------------------------------
-- %Z% Unit Name:        Moving_Model_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   19 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
---------------------------------------------------------------------
--
-- Purpose:
--  This package contains declarations of types which are (a) used to
--  define moving models, and (b) sent by more than one segment.
--
-- Adaptation:
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Moving_Model_Types is
--
--**********
--*
--/ 10.1.6         Moving Model Types
--*
--**********
--
--   used by FD,EW,WPN,PHC,NAV,RDR,VIS,FS,IOS

Bytes : constant := 8;

--
-- Moving_Model_Unique_Number and consequently moving model creation
-- shall be limited to the ranges specified below for the associated
-- modules:
--           Weapons   : Range 1..6      (Bombs)
--           EW        : Range 7..9      (Chaff, Flares, Platforms)
```

```
--            IOS       : Range 10..10   (Companion Aircraft)
--

--***************************
--*
--/ 10.1.6.1       Aircraft/Simulator Specific Moving Model Types
--*
--***************************

Maximum_Number_Of_Moving_Models     : constant := 10;
Maximum_Number_Of_Articulated_Parts : constant := 10;
Maximum_Number_Of_Model_Lights      : constant := 10;
Maximum_Number_Of_Emitters          : constant := 10;
Maximum_Number_Of_Chaff             : constant := 5;
Maximum_Number_Of_Flares            : constant := 5;

type Moving_Model is (
   KC_135R,
   Mk_82_Bomb,
   Chaff_TypeA,
   Chaff_TypeB,
   Flare_TypeA,
   Flare_TypeB,
   Threat_TypeA,
   Threat_TypeB);
for Moving_Model'size use 32;


--***************************
--*
--/ 10.1.6.2       Aircraft/Simulator Reusable Moving Model Types
--*
--***************************

subtype Moving_Model_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Moving_Models;

subtype Articulated_Part_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Articulated_Parts;

subtype Model_Lighting_Count is Base_Types.Unsigned_Integer_32
```

```
   range 1..Maximum_Number_Of_Model_Lights;

subtype Emitter_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Emitters;

subtype Chaff_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Chaff;

subtype Flare_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Flares;

subtype Chaff_And_Flare_Count is Base_Types.Unsigned_Integer_32
   range 1..(Maximum_Number_Of_Chaff +
   Maximum_Number_Of_Flares);

subtype Model_Unique_Number is Base_Types.Signed_Integer_32
     range 1..Maximum_Number_Of_Moving_Models;

type Moving_Model_ID is
 record
   Unique_Number : Model_Unique_Number;
   Kind          : Moving_Model;
end record;
for Moving_Model_ID use
 record
   Unique_Number at 0 range 0..31;
   Kind          at 4 range 0..31;
end record;
Moving_Model_ID_Size : constant := 8 * Bytes;
for Moving_Model_ID'size use Moving_Model_ID_Size;

type Moving_Model_Dynamic_Data is
 record
   ID                   : Moving_Model_ID;
   Position             : Engineering_Units.Earth_Position_Components;
   Velocity             : Engineering_Units.Earth_Velocity_Components;
   Acceleration         : Engineering_Units.Earth_Acceleration_Components;
   Attitude             : Engineering_Units.Angular_Position_Components;
   Angular_Velocity     : Engineering_Units.Angular_Velocity_Components;
   Angular_Acceleration : Engineering_Units.Angular_Acceleration_Components;
end record;
```

```
for Moving_Model_Dynamic_Data use
 record
   ID                     at  0 range 0..(8 * Bytes)-1;
   Position               at  8 range 0..(20 * Bytes)-1;
   Velocity               at 28 range 0..(12 * Bytes)-1;
   Acceleration           at 40 range 0..(12 * Bytes)-1;
   Attitude               at 52 range 0..(12 * Bytes)-1;
   Angular_Velocity       at 64 range 0..(12 * Bytes)-1;
   Angular_Acceleration   at 76 range 0..(12 * Bytes)-1;
end record;
Moving_Model_Dynamic_Data_Size : constant := 88 * Bytes;
for Moving_Model_Dynamic_Data'size use
   Moving_Model_Dynamic_Data_Size;

type Moving_Model_Deactivation is
 record
   ID    : Moving_Model_ID;
   State : Base_Types.Sim_Boolean := Base_Types.False;
end record;
for Moving_Model_Deactivation use
 record
   ID    at 0 range 0..63;
   State at 8 range 0..7;
end record;
for Moving_Model_Deactivation'size use 10 * Bytes;

type Model_Damage is (
   Left_Wing,
   Right_Wing,
   Tail);
for Model_Damage'size use 32;

type Model_Damage_Data is
 record
   Damage_Location : Model_Damage;
   Damage_Severity : Engineering_Units.Percent;
end record;
for Model_Damage_Data use
 record
   Damage_Location at 0 range 0..31;
   Damage_Severity at 4 range 0..31;
```

```
end record;
Model_Damage_Data_Size : constant := 8 * Bytes;
for Model_Damage_Data'size use
Model_Damage_Data_Size;

type Model_Damage_Data_Array is
     array (Moving_Model_Count) of
     Model_Damage_Data;
for Model_Damage_Data_Array'size use 80 * Bytes;

type Battle_Damage_Record is
 record
   Number_Of_Damages           : Moving_Model_Count;
   Number_Of_Damages_In_Array  : Moving_Model_Count;
   Battle_Damage_Data          : Model_Damage_Data_Array;
end record;
for Battle_Damage_Record use
 record
   Number_Of_Damages           at 0 range 0..31;
   Number_Of_Damages_In_Array  at 4 range 0..31;
   Battle_Damage_Data          at 8 range 0..(80 * Bytes)-1;
end record;
for Battle_Damage_Record'size use 88 * Bytes;

--
-- Weapons damage assessment
--
type Scoring_Damage_Data is -- Moved from WPN
 record
   Damage           : Model_Damage_Data;
   Damage_Caused_By : Moving_Model_ID;
   Damage_Caused_To : Moving_Model_ID;
end record;

type Scoring is (
   Ownship_Refueling,
   Bomb_Drop);
for Scoring'size use 8;

type Scoring_Activation_Status is
 record
```

```
    Scoring_Desired : Scoring;
    State           : Base_Types.Discrete_State;
end record;

type Articulated_Part is
     (Landing_Gear);
for Articulated_Part'size use 32;

type Articulated_Part_Data is
 record
    Part     : Articulated_Part;
    Position : Engineering_Units.Normalized;
end record;
for Articulated_Part_Data use
 record
    Part     at 0 range 0..31;
    Position at 4 range 0..31;
end record;
Articulated_Part_Data_Size : constant := 8 * Bytes;
for Articulated_Part_Data'size use
    Articulated_Part_Data_Size;

type Articulated_Part_Data_Array is
     array (Articulated_Part_Count) of
     Articulated_Part_Data;
for Articulated_Part_Data_Array'size use 80 * Bytes;

type Articulated_Device_Data_Record is
 record
    Number_Of_Articulated_Parts : Articulated_Part_Count;
    Number_Of_Parts_In_Array    : Articulated_Part_Count;
    Articulated_Parts_Data      : Articulated_Part_Data_Array;
end record;
for Articulated_Device_Data_Record use
 record
    Number_Of_Articulated_Parts at 0 range 0..31;
    Number_Of_Parts_In_Array    at 4 range 0..31;
    Articulated_Parts_Data      at 8 range 0..(80 * Bytes)-1;
end record;
for Articulated_Device_Data_Record'size use 88 * Bytes;
```

```
type Model_Lighting is
     (Running_Lights);
Model_Lighting_Size : constant := 8;
for Model_Lighting'size use Model_Lighting_Size;

type Model_Lighting_Data_Array is
     array (Model_Lighting_Count) of
     Model_Lighting;
for Model_Lighting_Data_Array'size use 10 * Bytes;

type Model_Lighting_Data_Record is
 record
   Number_Of_Model_Lights    : Model_Lighting_Count;
   Number_Of_Lights_In_Array : Model_Lighting_Count;
   Model_Lighting_Data       : Model_Lighting_Data_Array;
end record;
for Model_Lighting_Data_Record use
 record
   Number_Of_Model_Lights    at 0 range 0..31;
   Number_Of_Lights_In_Array at 4 range 0..31;
   Model_Lighting_Data       at 8 range 0..(10 * Bytes)-1;
end record;
for Model_Lighting_Data_Record'size use 20 * Bytes;

type Emitter_Frequency_Data is
 record
   Frequency   : Engineering_Units.MHz;
   Power_Level : Engineering_Units.Decibel;
end record;
for Emitter_Frequency_Data use
 record
   Frequency   at 0 range 0..31;
   Power_Level at 4 range 0..31;
end record;
Emitter_Frequency_Data_Size : constant := 8 * Bytes;
for Emitter_Frequency_Data'size use
Emitter_Frequency_Data_Size;

type Emitter_Data_Array is array (Emitter_Count) of
     Emitter_Frequency_Data;
for Emitter_Data_Array'size use 80 * Bytes;
```

```
type Emitter_Data_Record is
 record
   Number_Of_Emitters               : Emitter_Count;
   Number_Of_Frequencies_In_Array   : Emitter_Count;
   Emitter_Frequencies              : Emitter_Data_Array;
end record;
for Emitter_Data_Record use
 record
   Number_Of_Emitters               at 0 range 0..31;
   Number_Of_Frequencies_In_Array   at 4 range 0..31;
   Emitter_Frequencies              at 8 range 0..(80 * Bytes)-1;
end record;
for Emitter_Data_Record'size use 88 * Bytes;

type Emitter_Unique_Data is
 record
   No_of_Emitters : Emitter_Count;
   ID             : Moving_Model_ID;
   Emitter_Data   : Emitter_Data_Record;
end record;

--
-- Chaff and Flares
--
type Chaff_Moving_Model_Unique_Data is
 record
   ID                  : Moving_Model_ID;
   Chaff_Cloud_Radius  : Engineering_Units.Feet;
   Chaff_Cloud_Density : Engineering_Units.Normalized;
   Chaff_Cloud_Slope   : Engineering_Units.Feet_Per_Feet;
end record;
for Chaff_Moving_Model_Unique_Data use
 record
   ID                  at  0 range 0..63;
   Chaff_Cloud_Radius  at  8 range 0..31;
   Chaff_Cloud_Density at 12 range 0..31;
   Chaff_Cloud_Slope   at 16 range 0..31;
end record;
for Chaff_Moving_Model_Unique_Data'size use 20 * Bytes;
```

```
type Chaff_Moving_Model_Unique_Data_Array is
     array (Chaff_Count) of
     Chaff_Moving_Model_Unique_Data;
for Chaff_Moving_Model_Unique_Data_Array'size use 100 * Bytes;

type Chaff_Moving_Model_Unique_Data_Record is
 record
   Number_Of_Chaff : Chaff_Count;
   Chaff_Data      : Chaff_Moving_Model_Unique_Data_Array;
end record;
for Chaff_Moving_Model_Unique_Data_Record use
 record
   Number_Of_Chaff at 0 range 0..31;
   Chaff_Data      at 4 range 0..(100 * Bytes)-1;
end record;
for Chaff_Moving_Model_Unique_Data_Record'size use 104 * Bytes;

type Flare_Moving_Model_Unique_Data is
 record
   ID               : Moving_Model_ID;
   Flare_Brightness : Engineering_Units.Lumens;
   Flare_Radius     : Engineering_Units.Feet;
end record;
for Flare_Moving_Model_Unique_Data use
 record
   ID               at  0 range 0..63;
   Flare_Brightness at  8 range 0..31;
   Flare_Radius     at 12 range 0..31;
end record;
for Flare_Moving_Model_Unique_Data'size use 16 * Bytes;

type Flare_Moving_Model_Unique_Data_Array is
     array (Flare_Count) of
     Flare_Moving_Model_Unique_Data;
for Flare_Moving_Model_Unique_Data_Array'size use 80 * Bytes;

type Flare_Moving_Model_Unique_Data_Record is
 record
   Number_Of_Flares : Flare_Count;
   Flare_Data       : Flare_Moving_Model_Unique_Data_Array;
end record;
```

```
for Flare_Moving_Model_Unique_Data_Record use
 record
   Number_Of_Flares at 0 range 0..31;
   Flare_Data       at 4 range 0..(80 * Bytes)-1;
end record;
for Flare_Moving_Model_Unique_Data_Record'size use 84 * Bytes;

type Chaff_And_Flare_Moving_Model_Data_Array is
   array (Chaff_And_Flare_Count) of Moving_Model_Dynamic_Data;
Chaff_And_Flare_Moving_Model_Data_Array_Size : constant :=
   Moving_Model_Dynamic_Data_Size *
   (Maximum_Number_Of_Chaff + Maximum_Number_Of_Flares);
for Chaff_And_Flare_Moving_Model_Data_Array'size use
   Chaff_And_Flare_Moving_Model_Data_Array_Size;

type Chaff_And_Flares_Moving_Model_Data is
 record
   No_Of_Chaff_And_Flares       : Moving_Model_Count;
  Chaff_And_Flare_Dynamic_Data : Chaff_And_Flare_Moving_Model_Data_Array;
                                   --RDR, VIS, IOS
end record;
for Chaff_And_Flares_Moving_Model_Data use
 record
   No_Of_Chaff_And_Flares       at 0 range 0..(4 * Bytes)-1;
   Chaff_And_Flare_Dynamic_Data at 4 range 0..(880 * Bytes)-1;
     --RDR, VIS, IOS
end record;
for Chaff_And_Flares_Moving_Model_Data'size use 884 * Bytes;

type Chaff_And_Flares_Detail_Data is
 record
   No_Of_Chaff        : Chaff_Count;
   No_Of_Flares       : Flare_Count;
   Chaff_Unique_Data  : Chaff_Moving_Model_Unique_Data_Record;
                        --RDR, VIS, IOS
   Flares_Unique_Data : Flare_Moving_Model_Unique_Data_Record;
                        --RDR, VIS, IOS
end record;
for Chaff_And_Flares_Detail_Data use
 record
   No_Of_Chaff        at   0 range 0..(4 * Bytes)-1;
```

```
    No_Of_Flares        at   4 range 0..(4 * Bytes)-1;
    Chaff_Unique_Data   at   8 range 0..(104 * Bytes)-1;
    Flares_Unique_Data at 112 range 0..(84 * Bytes)-1;
end record;
for Chaff_And_Flares_Detail_Data'size use 196 * Bytes;

--
-- Decoy unique data (sent by EW and IOS)
--
type Decoy_Moving_Model_Unique_Data is
 record
    No_of_Emitters : Emitter_Count;
    ID             : Moving_Model_ID;
    Emitter_Data   : Emitter_Data_Record;
 end record;
for Decoy_Moving_Model_Unique_Data use
 record
    No_of_Emitters at  0 range 0..(4 * Bytes)-1;
    ID             at  4 range 0..(8 * Bytes)-1;
    Emitter_Data   at 12 range 0..(88 * Bytes)-1;
 end record;
for Decoy_Moving_Model_Unique_Data'size use 100 * Bytes;

--
-- Thunderstorm
--
type Thunderstorm_Moving_Model_Unique_Data is
 record
    ID              : Moving_Model_ID;
    Storm_Intensity : Engineering_Units.Zero_To_Ten;
end record;
for Thunderstorm_Moving_Model_Unique_Data use
 record
    ID              at 0 range 0..63;
    Storm_Intensity at 8 range 0..31;
end record;
for Thunderstorm_Moving_Model_Unique_Data'size use 12 * Bytes;

--
-- Platform unique data
--
```

```
type Platform_Moving_Model_Unique_Data is
 record
   No_Of_Emitters              : Emitter_Count;
   No_Of_Damages               : Moving_Model_Count;
   No_Of_Lights                : Model_Lighting_Count;
   No_Of_Articulated_Devices : Articulated_Part_Count;
   ID                          : Moving_Model_ID;
   Emitter_Data                : Emitter_Data_Record;
   IFF_ID                      : Global_Message_Types.IFF_Data;
   Battle_Damage_Data          : Battle_Damage_Record;
   Lighting_Data               : Model_Lighting_Data_Record;
   Articulated_Device_Data     : Articulated_Device_Data_Record;
  Weapon_Load_Status       : Global_Message_Types.Weapon_Station_Loading;
end record;
for Platform_Moving_Model_Unique_Data use
 record
   No_Of_Emitters              at   0 range 0..(4 * Bytes)-1;
   No_Of_Damages               at   4 range 0..(4 * Bytes)-1;
   No_Of_Lights                at   8 range 0..(4 * Bytes)-1;
   No_Of_Articulated_Devices at  12 range 0..(4 * Bytes)-1;
   ID                          at  16 range 0..(8 * Bytes)-1;
   Emitter_Data                at  24 range 0..(88 * Bytes)-1;
   IFF_ID                      at 112 range 0..(8 * Bytes)-1;
   Battle_Damage_Data          at 120 range 0..(88 * Bytes)-1;
   Lighting_Data               at 208 range 0..(20 * Bytes)-1;
   Articulated_Device_Data     at 228 range 0..(88 * Bytes)-1;
   Weapon_Load_Status          at 316 range 0..(8 * Bytes)-1;
end record;
for Platform_Moving_Model_Unique_Data'size use 324 * Bytes;

type Tanker_Moving_Model_Unique_Data is
 record
   No_Of_Emitters                      : Emitter_Count;
   No_Of_Damages                       : Moving_Model_Count;
   No_Of_Lights                        : Model_Lighting_Count;
   No_Of_Articulated_Devices           : Articulated_Part_Count;
   Tanker_Platform_Data              : Platform_Moving_Model_Unique_Data;
   Boom_Drogue_Connected               : Base_Types.Sim_Boolean;
   Boom_Drogue_Position_Wrt_Ownship :
Engineering_Units.Linear_Position_Components;
   Tanker_Fuel_Quantity                : Engineering_Units.Pounds;
   Tanker_Off_Load_Rate                : Engineering_Units.Lbs_Per_Hour;
```

```
end record;
for Tanker_Moving_Model_Unique_Data use
 record
   No_Of_Emitters                   at   0 range 0..(4 * Bytes)-1;
   No_Of_Damages                    at   4 range 0..(4 * Bytes)-1;
   No_Of_Lights                     at   8 range 0..(4 * Bytes)-1;
   No_Of_Articulated_Devices        at  12 range 0..(4 * Bytes)-1;
   Tanker_Platform_Data             at  16 range 0..(324 * Bytes)-1;
   Boom_Drogue_Connected            at 340 range 0..(1 * Bytes)-1;
   -- 3 spare bytes
   Boom_Drogue_Position_Wrt_Ownship at 344 range 0..(12 * Bytes)-1;
   Tanker_Fuel_Quantity             at 356 range 0..(4 * Bytes)-1;
   Tanker_Off_Load_Rate             at 360 range 0..(4 * Bytes)-1;
end record;
for Tanker_Moving_Model_Unique_Data'size use 364 * Bytes;

type Moving_Model_IFF_Data is
 record
   ID     : Moving_Model_ID;
   IFF_ID : Global_Message_Types.IFF_Data;
end record;
for Moving_Model_IFF_Data use
 record
   ID     at 0 range 0..63;
   IFF_ID at 8 range 0..63;
end record;
for Moving_Model_IFF_Data'size use 16 * Bytes;

end Moving_Model_Types;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Service_Function_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  19 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--   The Service Functions include: Radar Database, Visual Database,
--   Spatial Relations and Occulting.  These functions may be performed
--   by ENV, RDR or VIS.  This package declares types used in Service
--   Function messages.  Note: at this time no types are defined for
--   Radar Database or Visual Database.
--
-- Adaptation:
--   All the Service Function types are reusable, and should not require
--   any modification under normal circumstances.  It may be necessary
--   to define types for Radar Database or Visual Database messages,
--   should this be required on a given simulator.
--
with Control_Types;
with Engineering_Units;
with Moving_Model_Types;
--
package Service_Function_Types is
--
--*************************************************************
--*                                                           *
--/ 10.1.7          Service Function Types
--*                                                           *
--*************************************************************

Bytes : constant := 8;

--**********
```

```
--*
--/ 10.1.7.1        Radar Database
--*
--**********

-- NONE


--**********
--*
--/ 10.1.7.2        Visual Database
--*
--**********

-- NONE


--**********
--*
--/ 10.1.7.3        Spatial Relations
--*
--**********
--
--   Position Range used by NAV, VIS, RDR
--   Terrain Height used by ENV, NAV, VIS, RDR, FD, PHC, WPN

type Position_Range_Update is
 record
  Cursor_Position            : Engineering_Units.Earth_Position_Compo ents:
   Range_To_Cursor              : Engineering_Units.Feet;
   Bearing_To_Cursor            : Engineering_Units.Radians;
   Elevation_To_Cursor          : Engineering_Units.Radians;
   Range_Update_Time_Validity : Control_Types.Frame_Number;
end record;
for Position_Range_Update use
 record
   Cursor_Position              at  0 range 0..(20 * Bytes)-1;
   Range_To_Cursor              at 20 range 0..(4 * Bytes)-1;
   Bearing_To_Cursor            at 24 range 0..(4 * Bytes)-1;
   Elevation_To_Cursor          at 28 range 0..(4 * Bytes)-1;
   Range_Update_Time_Validity  at 32 range 0..(4 * Bytes)-1;
```

```
end record;
for Position_Range_Update'size use 36 * Bytes;

type Groundspeed_Update is
 record
   Doppler_Velocity             : Engineering_Units.Earth_Velocity_Components;
   Velocity_Update_Time_Validity : Control_Types.Frame_Number;
end record;
for Groundspeed_Update use
 record
   Doppler_Velocity              at  0 range 0..(12 * Bytes)-1;
   Velocity_Update_Time_Validity at 12 range 0..(4 * Bytes)-1;
end record;
for Groundspeed_Update'size use 16 * Bytes;

type Moving_Model_Terrain_Height is
 record
   Moving_Model          : Moving_Model_Types.Moving_Model_ID;
   Height_Above_Terrain : Engineering_Units.Feet;
end record;
for Moving_Model_Terrain_Height use
 record
   Moving_Model          at 0 range 0..(8 * Bytes)-1;
   Height_Above_Terrain at 8 range 0..(4 * Bytes)-1;
end record;
for Moving_Model_Terrain_Height'size use 12 * Bytes;

type Moving_Model_Terrain_Height_Array is
     array (Moving_Model_Types.Moving_Model_Count) of
Moving_Model_Terrain_Height;
for Moving_Model_Terrain_Height_Array'size use 120 * Bytes;

type Moving_Model_Terrain_Data is
 record
   Number_Of_Moving_Models      : Moving_Model_Types.Moving_Model_Count;
   Number_Of_Elements_In_Array  : Moving_Model_Types.Moving_Model_Count;
   Moving_Model_Terrain_Heights : Moving_Model_Terrain_Height_Array;
end record;
for Moving_Model_Terrain_Data use
 record
   Number_Of_Moving_Models      at 0 range 0..(4 * Bytes)-1;
```

```
   Number_Of_Elements_In_Array  at 4 range 0..(4 * Bytes)-1;
   Moving_Model_Terrain_Heights at 8 range 0..(120 * Bytes)-1;
end record;
for Moving_Model_Terrain_Data'size use 128 * Bytes;

type Earth_Surface is (
   Land,
   Sea);
Earth_Surface_Size : constant := 2 * Bytes;
for Earth_Surface'size use Earth_Surface_Size;

type Ownship_Height_Above_Terrain is
record
   Ownship_Height_Above_Terrain : Engineering_Units.Feet;
                                  -- RDR, WPN, PHC, VIS, FD
   Ownship_Over_Land_Or_Sea     : Earth_Surface;
                                  -- VIS
end record;
for Ownship_Height_Above_Terrain use
record
   Ownship_Height_Above_Terrain at 0 range 0..(4 * Bytes)-1;
   Ownship_Over_Land_Or_Sea     at 4 range 0..(2 * Bytes)-1;
end record;
for Ownship_Height_Above_Terrain'size use 6 * Bytes;

type Moving_Models_Height_Above_Terrain is
record
  No_Of_Models                        : Moving_Model_Types.Moving_Model_Count;
   Moving_Models_Height_Above_Terrain : Moving_Model_Terrain_Data;
end record;
for Moving_Models_Height_Above_Terrain use
record
   No_Of_Models                       at 0 range 0..(4 * Bytes)-1;
   Moving_Models_Height_Above_Terrain at 4 range 0..(128 * Bytes)-1;
end record;
for Moving_Models_Height_Above_Terrain'size use 132 * Bytes;


--***********
--*
--/ 10.1.7.4      Occulting
```

```
--*
--**********
--
--   used by ENV, NAV, EW, RDR, VIS

type Occult_Status is (Occulted, Visible);
for Occult_Status'size use 16;

type Occulting_Data is
 record
   Occult_Status_Of_Moving_Model : Moving_Model_Types.Moving_Model_ID;
   Occulting_With_Respect_To     : Moving_Model_Types.Moving_Model_ID;
   Current_Occulting_Status      : Occult_Status;
end record;
for Occulting_Data use
 record
   Occult_Status_Of_Moving_Model at  0 range 0..(8 * Bytes)-1;
   Occulting_With_Respect_To     at  8 range 0..(8 * Bytes)-1;
   Current_Occulting_Status      at 16 range 0..(2 * Bytes)-1;
end record;
for Occulting_Data'size use 20 * Bytes;

type Occulting_Data_Change_Array is
     array (Moving_Model_Types.Moving_Model_Count) of Occulting_Data;
for Occulting_Data_Change_Array'size use 200 * Bytes;

type Occulting_Status_Update is
 record
   Number_Of_Changes          : Moving_Model_Types.Moving_Model_Count;
   Number_Of_Array_Elements   : Moving_Model_Types.Moving_Model_Count;
   Occulting_Changes          : Occulting_Data_Change_Array;
end record;
for Occulting_Status_Update use
 record
   Number_Of_Changes        at 0 range 0..(4 * Bytes)-1;
   Number_Of_Array_Elements at 4 range 0..(4 * Bytes)-1;
   Occulting_Changes        at 8 range 0..(200 * Bytes)-1;
end record;
for Occulting_Status_Update'size use 208 * Bytes;

end Service_Function_Types;
```

```
------------------------------------------------------------------------
-- %Z% Unit Name:       Flight_Station_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  3 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Flight_Station segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Flight_Station_Output_Interface_Types is
--
--**************************************************************
--*                                                            *
--/ 10.2   Flight Station Output Interface Types               *
--*                                                            *
--**************************************************************

--**************************************
```

```
--*
--/ 10.2.1   Aircraft/Simulator Specific Flight Station Types
--*
--*******************************************

--
-- Discrete Inputs for each segment
--

type Environment_DIs is
   (TBD);
for Environment_DIs'size use 8;

type Electronic_Warfare_DIs is
   (TBD);
for Electronic_Warfare_DIs'size use 8;

type Flight_Controls_DIs is (
   Alternate_Flaps_Sw,
   Leading_Edge_Flaps_Sw,
   Trim_Disconnect_Sw,
   Manual_Pitch_Override_Sw,
   Stores_Configuration_Sw,
   Landing_Gear_Handle_Up,
   Left_Wing_Down_Trim_Sw,
   Nose_Down_Trim_Sw,
   Right_Wheel_Down_Trim_Sw,
   Nose_Up_Trim_Sw,
   Nosewheel_Steering_Button);
for Flight_Controls_DIs'size use 8;

type Flight_Dynamics_DIs is
   (TBD);
for Flight_Dynamics_DIs'size use 8;

type Flight_Station_DIs is (
   Anti_Skid_Switch,                     -- FC
   Landing_Gear_Down_Permission_Sw,      -- FC
   Landing_Gear_Handle_Up,               -- FC
   Alternate_Gear_Handle,                -- FC
   Alternate_Gear_Reset,                 -- FC
```

```
    Stores_Configuration_Sw,                -- FC
    Alternate_Flaps_Sw,                     -- FC
    Nosewheel_Steering_Button,              -- FC
    Nose_Up_Trim_Sw,                        -- FC
    Nose_Down_Trim_Sw,                      -- FC
    Left_Wing_Down_Trim_Sw,                 -- FC
    Right_Wing_Down_Trim_Sw,                -- FC
    Trim_Disconnect_Sw,                     -- FC
    Manual_Pitch_Override_Sw,               -- FC
    Leading_Edge_Flaps_Sw,                  -- FC
    Speedbrake_Fwd_Sw,                      -- FC
    Speedbrake_Aft_Sw,                      -- FC
    UFC_ILS_Mode_Flag,                      -- NAV
    Radar_Bomb_Scoring_Selected);           -- WPN
for Flight_Station_DIs'size use 8;

type Flight_Station_Analog is (
    Engine_Throttle_Position,               -- FC, PRO
    Pitch_Trim_Wheel,                       -- FC
    Roll_Trim_Wheel,                        -- FC
    Yaw_Trim_Knob);                         -- FC
for Flight_Station_Analog'size use 8;

type IOS_DIs is
    (TBD);
for IOS_DIs'size use 8;

type Navigation_DIs is (
    Cni_Backup_Switch,                      --NAV
    ILS_On_Switch,                          --NAV
    TACAN_On_Switch,                        --NAV
    Landing_Gear_Handle_Up,                 --NAV
    Caging_Caged_Switch,                    --NAV
    Alternate_Flaps_Switch,                 --NAV
    Leading_Edge_Flaps_Switch);             --NAV
for Navigation_DIs'size use 8;

type Physical_Cues_DIs is
    (TBD);
for Physical_Cues_DIs'size use 8;
```

```
type Propulsion_DIs is (
   Max_Power_Off_Switch,                     --PRO
   Throttle_At_Max_Ab_Detent_No,             --PRO
   Throttle_At_Mil_Detent_No,                --PRO
   Throttle_At_Idle_Detent_No,               --PRO
   Throttle_At_Off_Detent_No,                --PRO
   Landing_Gear_Handle_Up,                   --PRO
   BUC_Gnd_Test_Test_Switch);                --PRO
for Propulsion_DIs'size use 8;

type Radar_DIs is
   (TBD);
for Radar_DIs'size use 8;

type Visual_DIs is
   (TBD);
for Visual_DIs'size use 8;

type Weapons_DIs is (
   Emergency_Stores_Jettison_Switch,         --WPN
   Weapon_Release_Switch);                   --WPN
for Weapons_DIs'size use 8;

-- Multi-position switches
type Speedbrake_Position is (
   Neutral,
   Pushed_In,
   Pulled_Out); -- FC

type Parking_Brake_Position is (-- FC
   Off,
   Anti_Skid,
   On);

type Master_Arm_Switch is (
   Off,
   Arm,
   Simulate);  -- WPN

type JFS_Start_Switch is (-- PRO
   Start1,
```

```
   Start2,
   Off);

type EEC_BUC_Switch is (-- PRO
   Off,
   EEC,
   BUC);

type Starting_Fuel_Switch is (-- PRO
   Lean,
   Auto_Lean,
   Rich);

type Ralt_Power is (-- NAV
   Off,
   Stby,
   On);

type INS_Mode is (-- NAV
   Off,
   Att,
   Cal,
   NAV,
   Norm,
   Stor_Hdg);

type Instrument_Mode is (-- NAV
   ILS_NAV,
   NAV,
   TCN,
   ILS_TCN);

type Altimeter_Mode is (-- NAV
   Neutral,
   Pneu,
   Elec);

type TACAN_Function is (-- NAV
   T_R,
   Rec,
   A_A_Tr);
```

```
type TACAN_Band is (-- NAV
   X,
   Y);

-- Dummy type to ease future tailoring
type To_Be_Determined is (TBD);
for To_Be_Determined'size use 32;


--*****************************************
--*
--/ 10.2.2   Aircraft/Simulator Reusable Flight Station Types
--*
--*****************************************

type Flight_Station_Analog_Data_Array is
     array (Flight_Station_Analog) of Engineering_Units.Normalized;
-- where : 0 = lowest possibile position
--         1 = highest possibile position

----------------------------------------------------------------------
--
-- Discrete inputs for each segment
--
----------------------------------------------------------------------

-- ENV
Number_Of_Environment_DIs : constant :=
   Environment_DIs'pos (Environment_DIs'last) -
   Environment_DIs'pos (Environment_DIs'first) + 1;

subtype Environment_DI_Count is
   Base_Types.Unsigned_Integer_32 range 1..Number_Of_Environment_DIs;

type Environment_DI_And_State is
 record
   Name  : Environment_DIs;
   State : Base_Types.Discrete_State;
 end record;
```

```
type Environment_DI_And_State_Array is array (
   Environment_DI_Count) of
   Environment_DI_And_State;

-- EW
Number_Of_Electronic_Warfare_DIs : constant :=
   Electronic_Warfare_DIs'pos (Electronic_Warfare_DIs'last) -
   Electronic_Warfare_DIs'pos (Electronic_Warfare_DIs'first) + 1;

subtype Electronic_Warfare_DI_Count is
  Base_Types.Unsigned_Integer_32 range 1..Number_Of_Electronic_Warfare_DIs;

type Electronic_Warfare_DI_And_State is
 record
   Name  : Electronic_Warfare_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Electronic_Warfare_DI_And_State_Array is array (
   Electronic_Warfare_DI_Count) of
   Electronic_Warfare_DI_And_State;

-- FC
Number_Of_Flight_Controls_DIs : constant :=
   Flight_Controls_DIs'pos (Flight_Controls_DIs'last) -
   Flight_Controls_DIs'pos (Flight_Controls_DIs'first) + 1;

subtype Flight_Controls_DI_Count is
  Base_Types.Unsigned_Integer_32 range 1..Number_Of_Flight_Controls_DIs;

type Flight_Controls_DI_And_State is
 record
   Name  : Flight_Controls_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Flight_Controls_DI_And_State_Array is array (
   Flight_Controls_DI_Count) of
   Flight_Controls_DI_And_State;

-- FD
```

```
Number_Of_Flight_Dynamics_DIs : constant :=
   Flight_Dynamics_DIs'pos (Flight_Dynamics_DIs'last) -
   Flight_Dynamics_DIs'pos (Flight_Dynamics_DIs'first) + 1;

subtype Flight_Dynamics_DI_Count is
  Base_Types.Unsigned_Integer_32 range 1..Number_Of_Flight_Dynamics_DIs;

type Flight_Dynamics_DI_And_State is
 record
   Name  : Flight_Dynamics_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Flight_Dynamics_DI_And_State_Array is array (
   Flight_Dynamics_DI_Count) of
   Flight_Dynamics_DI_And_State;

-- FS
Number_Of_Flight_Station_DIs : constant :=
   Flight_Station_DIs'pos (Flight_Station_DIs'last) -
   Flight_Station_DIs'pos (Flight_Station_DIs'first) + 1;

subtype Flight_Station_DI_Count is
   Base_Types.Signed_Integer_32 range 1..Number_Of_Flight_Station_DIs;

type Flight_Station_DI_And_State is
record
   Name  : Flight_Station_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Flight_Station_DI_And_State_Array is array (
   Flight_Station_DI_Count) of
   Flight_Station_DI_And_State;

-- IOS
Number_Of_IOS_DIs : constant :=
   IOS_DIs'pos (IOS_DIs'last) -
   IOS_DIs'pos (IOS_DIs'first) + 1;

subtype IOS_DI_Count is
```

```
   Base_Types.Unsigned_Integer_32 range 1..Number_Of_IOS_DIs;

type IOS_DI_And_State is
 record
   Name  : IOS_DIs;
   State : Base_Types.Discrete_State;
 end record;

type IOS_DI_And_State_Array is array (
   IOS_DI_Count) of
   IOS_DI_And_State;

-- NAV
Number_Of_Navigation_DIs : constant :=
   Navigation_DIs'pos (Navigation_DIs'last) -
   Navigation_DIs'pos (Navigation_DIs'first) + 1;

subtype Navigation_DI_Count is
   Base_Types.Signed_Integer_32 range 1..Number_Of_Navigation_DIs;

type Navigation_DI_And_State is
 record
   Name  : Navigation_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Navigation_DI_And_State_Array is array (
   Navigation_DI_Count) of
   Navigation_DI_And_State;

-- PHC
Number_Of_Physical_Cues_DIs : constant :=
   Physical_Cues_DIs'pos (Physical_Cues_DIs'last) -
   Physical_Cues_DIs'pos (Physical_Cues_DIs'first) + 1;

subtype Physical_Cues_DI_Count is
   Base_Types.Unsigned_Integer_32 range 1..Number_Of_Physical_Cues_DIs;

type Physical_Cues_DI_And_State is
 record
   Name  : Physical_Cues_DIs;
```

```
   State : Base_Types.Discrete_State;
 end record;

type Physical_Cues_DI_And_State_Array is array (
   Physical_Cues_DI_Count) of
   Physical_Cues_DI_And_State;

-- PRO
Number_Of_Propulsion_DIs : constant :=
   Propulsion_DIs'pos (Propulsion_DIs'last) -
   Propulsion_DIs'pos (Propulsion_DIs'first) + 1;

subtype Propulsion_DI_Count is
   Base_Types.Signed_Integer_32 range 1..Number_Of_Propulsion_DIs;

type Propulsion_DI_And_State is
 record
   Name  : Propulsion_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Propulsion_DI_And_State_Array is array (
   Propulsion_DI_Count) of
   Propulsion_DI_And_State;

-- RDR
Number_Of_Radar_DIs : constant :=
   Radar_DIs'pos (Radar_DIs'last) -
   Radar_DIs'pos (Radar_DIs'first) + 1;

subtype Radar_DI_Count is
   Base_Types.Unsigned_Integer_32 range 1..Number_Of_Radar_DIs;

type Radar_DI_And_State is
 record
   Name  : Radar_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Radar_DI_And_State_Array is array (
   Radar_DI_Count) of
```

```
   Radar_DI_And_State;

-- VIS
Number_Of_Visual_DIs : constant :=
   Visual_DIs'pos (Visual_DIs'last) -
   Visual_DIs'pos (Visual_DIs'first) + 1;

subtype Visual_DI_Count is
   Base_Types.Unsigned_Integer_32 range 1..Number_Of_Visual_DIs;

type Visual_DI_And_State is
 record
   Name  : Visual_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Visual_DI_And_State_Array is array (
   Visual_DI_Count) of
   Visual_DI_And_State;

-- WPN
Number_Of_Weapons_DIs : constant :=
   Weapons_DIs'pos (Weapons_DIs'last) -
   Weapons_DIs'pos (Weapons_DIs'first) + 1;

subtype Weapons_DI_Count is
   Base_Types.Unsigned_Integer_32 range 1..Number_Of_Weapons_DIs;

type Weapons_DI_And_State is
 record
   Name  : Weapons_DIs;
   State : Base_Types.Discrete_State;
 end record;

type Weapons_DI_And_State_Array is array (
   Weapons_DI_Count) of
   Weapons_DI_And_State;


-----------------------------------------------------------------------------
--
```

```
-- Outputs from flight station to other segments (not cockpit controls)
--
---------------------------------------------------------------------------
type Generator_Electrical_Status is
 record
   Operating_State      : Base_Types.Discrete_State;
   Total_Electric_Load : Engineering_Units.Amperes;
   Output_Voltage       : Engineering_Units.kVA;
   Output_Frequency     : Engineering_Units.Hertz;
 end record;

type Generator_Electrical_Status_Array is
   array (Global_Message_Types.Aircraft_Electrical_Generator) of
   Generator_Electrical_Status;

type Generator_Drag_Torque_Array is
   array (Global_Message_Types.Aircraft_Electrical_Generator) of
   Engineering_Units.Ft_Lbs;

type Aircraft_Electrical_Bus_Voltage_Array is
   array (Global_Message_Types.Aircraft_Electrical_Bus) of
   Engineering_Units.kVA;

type Aircraft_Hydraulic_Reservoir_Data_Array is
   array (Global_Message_Types.Aircraft_Hydraulic_Reservoir) of
   Global_Message_Types.Fluid_Characteristics;

type Hydraulic_Pump_Drag_Torque_Array is
   array (Global_Message_Types.Aircraft_Hydraulic_Pump) of
   Engineering_Units.Ft_Lbs;

type Hydraulic_System_Pressure_Array is
   array (Global_Message_Types.Aircraft_Hydraulic_System) of
   Engineering_Units.PSI;

type Aircraft_Hydraulic_Component_Pressure_Array is
   array (Global_Message_Types.Aircraft_Hydraulic_Component) of
   Engineering_Units.PSI;

type Engine_Inlet_Fuel_Data is
 record
```

```
   Available_Engine_Fuel_Flow : Engineering_Units.Lbs_Per_Hour;
   Inlet_Fuel_Temperature     : Engineering_Units.Degrees_C;
   Inlet_Fuel_Pressure        : Engineering_Units.PSI;
 end record;

type Engine_Inlet_Fuel_Data_Array is
   array (Global_Message_Types.Aircraft_Engine) of
Engine_Inlet_Fuel_Data;

type Available_APU_Fuel_Flow_Array is
   array (Global_Message_Types.Aircraft_APU) of
   Engineering_Units.Lbs_Per_Hour;

type Fuel_Tank_Temperature_Array is
   array (Global_Message_Types.Aircraft_Fuel_Tank) of
   Engineering_Units.Degrees_C;

type Engine_Bleed_Air_Flow_Demand_Array is
   array (Global_Message_Types.Aircraft_Engine) of
   Engineering_Units.Ft3_Per_Min;

type APU_Bleed_Air_Flow_Demand_Array is
   array (Global_Message_Types.Aircraft_APU) of
   Engineering_Units.Ft3_Per_Min;

type Engine_Starting_Air_Pressure_Array is
   array (Global_Message_Types.Aircraft_Engine) of
   Engineering_Units.PSI;

type Air_Characteristics is
 record
   Quantity : Engineering_Units.Cubic_Feet;
   Pressure : Engineering_Units.PSI;
end record;

type Oxygen_System_Data_Array is
   array (Global_Message_Types.Aircraft_Oxygen_System) of
   Air_Characteristics;

type Aircraft_Pneumatic_Component_Pressure_Array is
   array (Global_Message_Types.Aircraft_Pneumatic_Component) of
   Engineering_Units.PSI;
```

```
--*****************************************
--*
--/ 10.2.3   Flight Station Segment Output Records
--*
--*****************************************


--***************Function:
--*
--/ 10.2.3.1   Electrical System
--*

type Electrical_System_Sixteenth_Rate is
 record
   Total_Bus_Load   : Global_Message_Types.Aircraft_Electrical_Bus_Load_Array;
                        --IOS
   Bus_Voltage        : Aircraft_Electrical_Bus_Voltage_Array;
                       --IOS,NAV,FD,FC,Electronic_Warfare,RDR,WPN,PRO,VIS
   Generator_Status : Generator_Electrical_Status_Array;
                        --IOS,PRO
 end record;

type Electrical_System_Quarter_Rate is
 record
   Generator_Drag_Torque : Generator_Drag_Torque_Array; --PRO
 end record;


--***************Function:
--*
--/ 10.2.3.2   Hydraulic System
--*

type Hydraulic_System_Sixteenth_Rate is
 record
   Hydraulic_Reservoir_Data :
     Aircraft_Hydraulic_Reservoir_Data_Array; -- IOS,FC
 end record;
```

```
type Hydraulic_System_Quarter_Rate is
 record
   Hydraulic_Pump_Drag_Torque     : Hydraulic_Pump_Drag_Torque_Array;
--PRO
   Hydraulic_Component_Pressures :
Aircraft_Hydraulic_Component_Pressure_Array;
     --IOS,NAV,FD,FC,Electronic_Warfare,RDR,WPN,PRO,VIS
   Hydraulic_System_Pressures     : Hydraulic_System_Pressure_Array; --
IOS
 end record;


--***************Function:
--*
--/ 10.2.3.3   Fuel Management System
--*

type Fuel_Management_System_Sixteenth_Rate is
 record
  Fuel_Tank_Quantities   : Global_Message_Types.Fuel_Tank_Quantity_Array;
                             --IOS,FD
   Boom_Fuel_Pressure      : Engineering_Units.PSI; --FD
   Fuel_Tank_Temperatures : Fuel_Tank_Temperature_Array;--IOS
 end record;

type Fuel_Management_System_Eighth_Rate is
 record
   Available_APU_Fuel_Flows : Available_APU_Fuel_Flow_Array;
    --PRO
   Fuel_Data_At_Engine_Inlets : Engine_Inlet_Fuel_Data; --PRO
 end record;


--***************Function:
--*
--/ 10.2.3.4   Pneumatic System
--*

type Pneumatic_System_Sixteenth_Rate is
 record
   Engine_Bleed_Air_Flow_Demands : Engine_Bleed_Air_Flow_Demand_Array;
--PRO
```

```
   APU_Bleed_Air_Flow_Demands     : APU_Bleed_Air_Flow_Demand_Array;
--PRO
   Engine_Starting_Air_Pressure  : Engine_Starting_Air_Pressure_Array;
--PRO
   Cabin_Altitude                : Engineering_Units.Feet; --IOS
   Cabin_Differential_Pressure   : Engineering_Units.PSI; --IOS
   Cabin_Rate_Of_Climb           : Engineering_Units.Ft_Per_Min; --IOS
   Pneumatic_Component_Pressures :
     Aircraft_Pneumatic_Component_Pressure_Array; --FC;
 end record;


--***************Function:
--*
--/ 10.2.3.5   Autochecklist System
--*

--NONE


--***************Function:
--*
--/ 10.2.3.6   Oxygen System
--*

type Oxygen_System_Sixteenth_Rate is
 record
   Oxygen_System_Data : Oxygen_System_Data_Array; --IOS
 end record;


--***************Function:
--*
--/ 10.2.3.7   Crew Station Interface
--*

type Crew_Station_Interface_Half_Rate is
 record
   Current_Analog_Data : Flight_Station_Analog_Data_Array;

--IOS,NAV,FD,FC,Electronic_Warfare,RDR,WPN,PRO,VIS,PHC
 end record;
```

```
type Electronic_Warfare_AI_Max_Rate is
 record
   TBD_Field_One   : To_Be_Determined;
   TBD_Field_Two   : To_Be_Determined;
   TBD_Field_Three : To_Be_Determined;
 end record;

type Flight_Controls_AI_Max_Rate is                    -- FC
 record
   Roll_Trim  : Engineering_Units.Signed_Normalized;
   Pitch_Trim : Engineering_Units.Signed_Normalized;
   Yaw_Trim   : Engineering_Units.Signed_Normalized;
 end record;

type Flight_Dynamics_AI_Max_Rate is
 record
   TBD_Field_One   : To_Be_Determined;
   TBD_Field_Two   : To_Be_Determined;
   TBD_Field_Three : To_Be_Determined;
 end record;

type IOS_AI_Max_Rate is
 record
   TBD_Field_One   : To_Be_Determined;
   TBD_Field_Two   : To_Be_Determined;
   TBD_Field_Three : To_Be_Determined;
 end record;

type Navigation_AI_Max_Rate is
 record
   Baro_Out           : Engineering_Units.Inches_Hg;
   Instrument_Heading : Engineering_Units.Degrees;
   Heading_Set_Knob   : Engineering_Units.Signed_Degrees;
   Course_Set_Knob    : Engineering_Units.Degrees;
   ADI_Pitch_Trim     : Engineering_Units.Signed_Degrees;
 end record;

type Physical_Cues_AI_Max_Rate is
 record
   TBD_Field_One   : To_Be_Determined;
```

```
    TBD_Field_Two   : To_Be_Determined;
    TBD_Field_Three : To_Be_Determined;
  end record;

type Propulsion_AI_Max_Rate is               -- PRO
  record
    Throttle_Position : Engineering_Units.Normalized;
  end record;

type Radar_AI_Max_Rate is
  record
    TBD_Field_One   : To_Be_Determined;
    TBD_Field_Two   : To_Be_Determined;
    TBD_Field_Three : To_Be_Determined;
  end record;

type Visual_AI_Max_Rate is
  record
    TBD_Field_One   : To_Be_Determined;
    TBD_Field_Two   : To_Be_Determined;
    TBD_Field_Three : To_Be_Determined;
  end record;

type Weapons_AI_Max_Rate is
  record
    TBD_Field_One   : To_Be_Determined;
    TBD_Field_Two   : To_Be_Determined;
    TBD_Field_Three : To_Be_Determined;
  end record;

-- SEND-ON-CHANGE

--
-- Discrete Inputs
--
type Environment_Discrete_Input_List is
  record
    Number_Of_DIs   : Environment_DI_Count;
    Discrete_Inputs : Environment_DI_And_State_Array;
  end record;
```

```
type Electronic_Warfare_Discrete_Input_List is
 record
   Number_Of_DIs   : Electronic_Warfare_DI_Count;
   Discrete_Inputs : Electronic_Warfare_DI_And_State_Array;
 end record;

type Flight_Controls_Discrete_Input_List is
record
   Number_Of_DIs   : Flight_Controls_DI_Count;
   Discrete_Inputs : Flight_Controls_DI_And_State_Array;
end record;

type Flight_Dynamics_Discrete_Input_List is
 record
   Number_Of_DIs   : Flight_Dynamics_DI_Count;
   Discrete_Inputs : Flight_Dynamics_DI_And_State_Array;
 end record;

type Flight_Station_Discrete_Input_List is
record
   Number_Of_DIs   : Flight_Station_DI_Count;
   Discrete_Inputs : Flight_Station_DI_And_State_Array;
 end record;

type IOS_Discrete_Input_List is
 record
   Number_Of_DIs   : IOS_DI_Count;
   Discrete_Inputs : IOS_DI_And_State_Array;
 end record;

type Navigation_Discrete_Input_List is
record
   Number_Of_DIs   : Navigation_DI_Count;
   Discrete_Inputs : Navigation_DI_And_State_Array;
end record;

type Physical_Cues_Discrete_Input_List is
 record
   Number_Of_DIs   : Physical_Cues_DI_Count;
   Discrete_Inputs : Physical_Cues_DI_And_State_Array;
 end record;
```

```
type Propulsion_Discrete_Input_List is
record
   Number_Of_DIs    : Propulsion_DI_Count;
   Discrete_Inputs : Propulsion_DI_And_State_Array;
end record;

type Radar_Discrete_Input_List is
 record
   Number_Of_DIs    : Radar_DI_Count;
   Discrete_Inputs . Radar_DI_And_State_Array;
 end record;

type Visual_Discrete_Input_List is
 record
   Number_Of_DIs    : Visual_DI_Count;
   Discrete_Inputs : Visual_DI_And_State_Array;
 end record;

type Weapons_Discrete_Input_List is
record
   Number_Of_DIs    : Weapons_DI_Count;
   Discrete_Inputs : Weapons_DI_And_State_Array;
end record;

-- see Global_Message_Types for definitions of Master_Mode,
-- SMS_Submode, A_G_Weapon_Delivery_Mode, Weapon_Station_Change,
-- and Stores_Station

-- Weapon station option
type Weapon_Station_Option is                    -- WPN
 record
   Station          : Global_Message_Types.Stores_Station;
   Profile          : Global_Message_Types.Weapon_Profile;
   Release          : Global_Message_Types.Release_Option;
   Release_Pulses   : Base_Types.Unsigned_Integer_8;
   Interval         : Engineering_Units.Feet;
   Arming           : Global_Message_Types.Arming_Option;
   Fuze             : Global_Message_Types.Fuze_Arming;
   Arming_Delay1    : Engineering_Units.Time_In_Seconds;
   Arming_Delay2    : Engineering_Units.Time_In_Seconds;
```

```
    Burst_Altitude   : Engineering_Units.Feet;
    Pull_Up_Range    : Engineering_Units.Feet;
    Time_Of_Fall     : Engineering_Units.Time_In_Seconds;
  end record;

-- Jettison
type Jettison_Station is  -- WPN
  record
    Jettison_Kind : Global_Message_Types.Jettison_Type;
    Station       : Global_Message_Types.Stores_Station;
  end record;


-- UFC Steerpoint
type UFC_Steerpoint_Selection is      -- NAV
  record
    Steerpoint_Number : Global_Message_Types.INS_Waypoints;
  end record;

-- TACAN data
type TACAN_Data is
  record
    Channel : Base_Types.Signed_Integer_16;    -- NAV
    Fun     : TACAN_Function;                  -- NAV
    Band    : TACAN_Band;                      -- NAV
  end record;

-- The following are defined above under Aircraft/Simulator
-- Specific types:
--   Speedbrake_Switch_Change_Of_State
--   Parking_Brake_Switch_Change_Of_State
--   Master_Arm_Switch_Change_Of_State
--   JFS_Start_Switch_Change_Of_State
--   EEC_BUC_Switch_Change_Of_State
--   Starting_Fuel_Switch_Change_Of_State
--   Ralt_Power_Switch_Change_Of_State
--   INS_Mode_Switch_Change_Of_State
--   Instrument_Mode_Switch_Change_Of_State
--   Altimeter_Mode_Switch_Change_Of_State
```

```
--***************Function:
--*
--/ 10.2.3.8   Flight Station Support
--*

-- See Control_Types for responses to IOS


--*****************************************
--*
--/ 10.2.4   Flight Station Representation Specs
--*
--*****************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes          : constant := 8; -- Bits per byte
   Byte_Size      : constant := 1 * Bytes;
   Halfword_Size  : constant := 2 * Bytes;
   Word_Size      : constant := 4 * Bytes;

   -- 10.2.3.1
   Aircraft_Electrical_Bus_Load_Array_Size : constant :=
      Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size;

   Aircraft_Electrical_Bus_Voltage_Array_Size : constant :=
      Global_Message_Types.Number_Of_Aircraft_Electrical_Busses *
Word_Size;
   for Aircraft_Electrical_Bus_Voltage_Array'size use
      Aircraft_Electrical_Bus_Voltage_Array_Size;

   for Generator_Electrical_Status use
    record
      Operating_State     at  0 range 0..Byte_Size-1;
      -- 3 bytes spare
      Total_Electric_Load at 1 * Word_Size/Bytes range 0..Word_Size-1;
      Output_Voltage      at 2 * Word_Size/Bytes range 0..Word_Size-1;
      Output_Frequency    at 3 * Word_Size/Bytes range 0..Word_Size-1;
    end record;
   Generator_Electrical_Status_Size : constant := 4 * Word_Size;
```

```
for Generator_Electrical_Status'size use
   Generator_Electrical_Status_Size;

Number_Of_Generators : constant :=
   Global_Message_Types.Aircraft_Electrical_Generator'pos (
   Global_Message_Types.Aircraft_Electrical_Generator'last) -
   Global_Message_Types.Aircraft_Electrical_Generator'pos (
   Global_Message_Types.Aircraft_Electrical_Generator'first) + 1;

Generator_Electrical_Status_Array_Size : constant :=
   Number_Of_Generators * Generator_Electrical_Status_Size;
for Generator_Electrical_Status_Array'size use
   Generator_Electrical_Status_Array_Size;

for Electrical_System_Sixteenth_Rate use
 record
   Total_Bus_Load
      at 0
      range 0..Aircraft_Electrical_Bus_Load_Array_Size-1;
   Bus_Voltage
      at Aircraft_Electrical_Bus_Load_Array_Size/Bytes
      range 0..Aircraft_Electrical_Bus_Voltage_Array_Size-1;
   Generator_Status
      at Aircraft_Electrical_Bus_Load_Array_Size/Bytes +
         Aircraft_Electrical_Bus_Voltage_Array_Size/Bytes
      range 0..Generator_Electrical_Status_Array_Size-1;
 end record;
for Electrical_System_Sixteenth_Rate'size use
   Aircraft_Electrical_Bus_Load_Array_Size +
   Aircraft_Electrical_Bus_Voltage_Array_Size +
   Generator_Electrical_Status_Array_Size;

Generator_Drag_Torque_Array_Size : constant :=
   Number_Of_Generators * Word_Size;
for Generator_Drag_Torque_Array'size use
   Generator_Drag_Torque_Array_Size;

for Electrical_System_Quarter_Rate use
 record
   Generator_Drag_Torque at 0 range 0..
      Generator_Drag_Torque_Array_Size-1;
```

```
 end record;
for Electrical_System_Quarter_Rate'size use
   Generator_Drag_Torque_Array_Size,


-- 10.2.3.2
Number_Of_Hydraulic_Reservoirs : constant :=
   Global_Message_Types.Aircraft_Hydraulic_Reservoir'pos (
   Global_Message_Types.Aircraft_Hydraulic_Reservoir'last) -
   Global_Message_Types.Aircraft_Hydraulic_Reservoir'pos (
   Global_Message_Types.Aircraft_Hydraulic_Reservoir'first) + 1;

Aircraft_Hydraulic_Reservoir_Data_Array_Size : constant :=
   Number_Of_Hydraulic_Reservoirs *
   Global_Message_Types.Fluid_Characteristics_Size;
for Aircraft_Hydraulic_Reservoir_Data_Array'size use
   Aircraft_Hydraulic_Reservoir_Data_Array_Size;

for Hydraulic_System_Sixteenth_Rate use
 record
   Hydraulic_Reservoir_Data at 0 range 0..
      Aircraft_Hydraulic_Reservoir_Data_Array_Size-1;
 end record;
for Hydraulic_System_Sixteenth_Rate'size use
   Aircraft_Hydraulic_Reservoir_Data_Array_Size;

Number_Of_Hydraulic_Pumps : constant :=
   Global_Message_Types.Aircraft_Hydraulic_Pump'pos (
   Global_Message_Types.Aircraft_Hydraulic_Pump'last) -
   Global_Message_Types.Aircraft_Hydraulic_Pump'pos (
   Global_Message_Types.Aircraft_Hydraulic_Pump'first) + 1;

Hydraulic_Pump_Drag_Torque_Array_Size : constant :=
   Number_Of_Hydraulic_Pumps * Word_Size;

for Hydraulic_Pump_Drag_Torque_Array'size use
   Hydraulic_Pump_Drag_Torque_Array_Size;

Number_Of_Hydraulic_Components : constant :=
   Global_Message_Types.Aircraft_Hydraulic_Component'pos (
   Global_Message_Types.Aircraft_Hydraulic_Component'last) -
```

```
   Global_Message_Types.Aircraft_Hydraulic_Component'pos (
   Global_Message_Types.Aircraft_Hydraulic_Component'first) + 1;

Aircraft_Hydraulic_Component_Pressure_Array_Size : constant :=
   Number_Of_Hydraulic_Components * Word_Size;
for Aircraft_Hydraulic_Component_Pressure_Array'size use
   Aircraft_Hydraulic_Component_Pressure_Array_Size;

Number_Of_Hydraulic_Systems : constant :=
   Global_Message_Types.Aircraft_Hydraulic_System'pos (
   Global_Message_Types.Aircraft_Hydraulic_System'last) -
   Global_Message_Types.Aircraft_Hydraulic_System'pos (
   Global_Message_Types.Aircraft_Hydraulic_System'first) + 1;

Hydraulic_System_Pressure_Array_Size : constant :=
   Number_Of_Hydraulic_Systems * Word_Size;
for Hydraulic_System_Pressure_Array'size use
   Hydraulic_System_Pressure_Array_Size;

for Hydraulic_System_Quarter_Rate use
 record
   Hydraulic_Pump_Drag_Torque
      at 0
      range 0..Hydraulic_Pump_Drag_Torque_Array_Size-1;
   Hydraulic_Component_Pressures
      at Hydraulic_Pump_Drag_Torque_Array_Size/Bytes
      range 0..Aircraft_Hydraulic_Component_Pressure_Array_Size-1;
   Hydraulic_System_Pressures
      at Hydraulic_Pump_Drag_Torque_Array_Size/Bytes +
         Aircraft_Hydraulic_Component_Pressure_Array_Size/Bytes
      range 0..Hydraulic_System_Pressure_Array_Size-1;
 end record;
for Hydraulic_System_Quarter_Rate'size use
   Hydraulic_Pump_Drag_Torque_Array_Size +
   Aircraft_Hydraulic_Component_Pressure_Array_Size +
   Hydraulic_System_Pressure_Array_Size;

-- 10.2.3.3
Fuel_Tank_Quantity_Array_Size : constant :=
   Global_Message_Types.Fuel_Tank_Quantity_Array_Size;
```

```
Fuel_Tank_Temperature_Array_Size : constant :=
   Global_Message_Types.Number_Of_Fuel_Tanks * Word_Size;
for Fuel_Tank_Temperature_Array'size use
   Fuel_Tank_Temperature_Array_Size;

for Fuel_Management_System_Sixteenth_Rate use
 record
   Fuel_Tank_Quantities
      at  0
      range 0..Fuel_Tank_Quantity_Array_Size-1;
   Boom_Fuel_Pressure
      at Fuel_Tank_Quantity_Array_Size/Bytes
      range 0..Word_Size-1;
   Fuel_Tank_Temperatures
      at Fuel_Tank_Quantity_Array_Size/Bytes +
         Word_Size/Bytes
      range 0..Fuel_Tank_Temperature_Array_Size-1;
 end record;
for Fuel_Management_System_Sixteenth_Rate'size use
   Fuel_Tank_Quantity_Array_Size +
   Word_Size +
   Fuel_Tank_Temperature_Array_Size;

Number_Of_APUs : constant :=
   Global_Message_Types.Aircraft_APU'pos (
   Global_Message_Types.Aircraft_APU'last) -
   Global_Message_Types.Aircraft_APU'pos (
   Global_Message_Types.Aircraft_APU'first) + 1;

Available_APU_Fuel_Flow_Array_Size : constant :=
   Number_Of_APUs * Word_Size;
for Available_APU_Fuel_Flow_Array'size use
   Available_APU_Fuel_Flow_Array_Size;

for Engine_Inlet_Fuel_Data use
 record
   Available_Engine_Fuel_Flow at 0 range 0..Word_Size-1;
   Inlet_Fuel_Temperature     at 4 range 0..Word_Size-1;
   Inlet_Fuel_Pressure        at 8 range 0..Word_Size-1;
 end record;
Engine_Inlet_Fuel_Data_Size : constant := 3 * Word_Size;
```

```
for Engine_Inlet_Fuel_Data'size use
   Engine_Inlet_Fuel_Data_Size;

Number_Of_Engines : constant :=
   Global_Message_Types.Aircraft_Engine'pos (
   Global_Message_Types.Aircraft_Engine'last) -
   Global_Message_Types.Aircraft_Engine'pos (
   Global_Message_Types.Aircraft_Engine'first) + 1;

Engine_Inlet_Fuel_Data_Array_Size : constant :=
   Number_Of_Engines * Engine_Inlet_Fuel_Data_Size;
for Engine_Inlet_Fuel_Data_Array'size use
   Engine_Inlet_Fuel_Data_Array_Size;

for Fuel_Management_System_Eighth_Rate use
 record
   Available_APU_Fuel_Flows
      at 0
      range 0..Available_APU_Fuel_Flow_Array_Size-1;
   Fuel_Data_At_Engine_Inlets
      at Available_APU_Fuel_Flow_Array_Size/Bytes
      range 0..Engine_Inlet_Fuel_Data_Array_Size-1;
end record;
for Fuel_Management_System_Eighth_Rate'size use
      Available_APU_Fuel_Flow_Array_Size +
      Engine_Inlet_Fuel_Data_Array_Size;

-- 10.2.3.4
Number_Of_Pneumatic_Components : constant :=
   Global_Message_Types.Aircraft_Pneumatic_Component'pos (
   Global_Message_Types.Aircraft_Pneumatic_Component'last) -
   Global_Message_Types.Aircraft_Pneumatic_Component'pos (
   Global_Message_Types.Aircraft_Pneumatic_Component'first) + 1;

Aircraft_Pneumatic_Component_Pressure_Array_Size : constant :=
   Number_Of_Pneumatic_Components * Word_Size;
for Aircraft_Pneumatic_Component_Pressure_Array'size use
   Aircraft_Pneumatic_Component_Pressure_Array_Size;

Engine_Bleed_Air_Flow_Demand_Array_Size : constant :=
   Number_Of_Engines * Word_Size;
```

```
for Engine_Bleed_Air_Flow_Demand_Array'size use
   Engine_Bleed_Air_Flow_Demand_Array_Size;

APU_Bleed_Air_Flow_Demand_Array_Size : constant :=
   Number_Of_APUs * Word_Size;
for APU_Bleed_Air_Flow_Demand_Array'size use
   APU_Bleed_Air_Flow_Demand_Array_Size;

Engine_Starting_Air_Pressure_Array_Size : constant :=
   Number_Of_Engines * Word_Size;
for Engine_Starting_Air_Pressure_Array'size use
   Engine_Starting_Air_Pressure_Array_Size;

for Pneumatic_System_Sixteenth_Rate use
 record
   Engine_Bleed_Air_Flow_Demands
      at  0
      range 0..Engine_Bleed_Air_Flow_Demand_Array_Size-1;
   APU_Bleed_Air_Flow_Demands
      at Engine_Bleed_Air_Flow_Demand_Array_Size/Bytes
      range 0..APU_Bleed_Air_Flow_Demand_Array_Size-1;
   Engine_Starting_Air_Pressure
      at Engine_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         APU_Bleed_Air_Flow_Demand_Array_Size/Bytes
      range 0..Engine_Starting_Air_Pressure_Array_Size-1;
   Cabin_Altitude
      at Engine_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         APU_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         Engine_Starting_Air_Pressure_Array_Size/Bytes
      range 0..Word_Size-1;
   Cabin_Differential_Pressure
      at Engine_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         APU_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         Engine_Starting_Air_Pressure_Array_Size/Bytes +
         Word_Size/Bytes
      range 0..Word_Size-1;
   Cabin_Rate_Of_Climb
      at Engine_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         APU_Bleed_Air_Flow_Demand_Array_Size/Bytes +
         Engine_Starting_Air_Pressure_Array_Size/Bytes +
         Word_Size/Bytes +
```

```
        Word_Size/Bytes
     range 0..Word_Size-1;
  Pneumatic_Component_Pressures
     at Engine_Bleed_Air_Flow_Demand_Array_Size/Bytes +
        APU_Bleed_Air_Flow_Demand_Array_Size/Bytes +
        Engine_Starting_Air_Pressure_Array_Size/Bytes +
        Word_Size/Bytes +
        Word_Size/Bytes +
        Word_Size/Bytes
     range 0..Aircraft_Pneumatic_Component_Pressure_Array_Size-1;
end record;
for Pneumatic_System_Sixteenth_Rate'size use
   Engine_Bleed_Air_Flow_Demand_Array_Size +
   APU_Bleed_Air_Flow_Demand_Array_Size +
   Engine_Starting_Air_Pressure_Array_Size +
   Word_Size +
   Word_Size +
   Word_Size +
   Aircraft_Pneumatic_Component_Pressure_Array_Size;

-- 10.2.3.6
Number_Of_Oxygen_Systems : constant :=
   Global_Message_Types.Aircraft_Oxygen_System'pos (
   Global_Message_Types.Aircraft_Oxygen_System'last) -
   Global_Message_Types.Aircraft_Oxygen_System'pos (
   Global_Message_Types.Aircraft_Oxygen_System'first) + 1;

for Air_Characteristics use
 record
   Quantity at 0 range 0..Word_Size-1;
   Pressure at 4 range 0..Word_Size-1;
end record;

Air_Characteristics_Size : constant := 2 * Word_Size;
for Air_Characteristics'size use Air_Characteristics_Size;

Oxygen_System_Data_Array_Size : constant :=
   Number_Of_Oxygen_Systems * Air_Characteristics_Size;
for Oxygen_System_Data_Array'size use
   Oxygen_System_Data_Array_Size;
```

```
for Oxygen_System_Sixteenth_Rate use
 record
   Oxygen_System_Data at 0 range 0..(16 * Bytes)-1;
 end record;
for Oxygen_System_Sixteenth_Rate'size use
   Oxygen_System_Data_Array_Size;

-- 10.2.3.7
Number_Of_Flight_Station_Analogs : constant :=
   Flight_Station_Analog'pos (Flight_Station_Analog'last) -
   Flight_Station_Analog'pos (Flight_Station_Analog'first) + 1;

Flight_Station_Analog_Data_Array_Size : constant :=
   Number_Of_Flight_Station_Analogs * Word_Size;
for Flight_Station_Analog_Data_Array'size use
   Flight_Station_Analog_Data_Array_Size;

for Crew_Station_Interface_Half_Rate use
 record
   Current_Analog_Data at 0 range 0..
      Flight_Station_Analog_Data_Array_Size-1;
 end record;
for Crew_Station_Interface_Half_Rate'size use
   Flight_Station_Analog_Data_Array_Size;

for Electronic_Warfare_AI_Max_Rate use
 record
   TBD_Field_One   at 0                     range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes   range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes   range 0..Word_Size-1;
 end record;
for Electronic_Warfare_AI_Max_Rate'size use 3 * Word_Size;

for Flight_Controls_AI_Max_Rate use
 record
   Roll_Trim        at 0                     range 0..Word_Size-1;
   Pitch_Trim       at 1 * Word_Size/Bytes   range 0..Word_Size-1;
   Yaw_Trim         at 2 * Word_Size/Bytes   range 0..Word_Size-1;
 end record;
for Flight_Controls_AI_Max_Rate'size use 3 * Word_Size;
```

```
for Flight_Dynamics_AI_Max_Rate use
 record
   TBD_Field_One   at 0                    range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for Flight_Dynamics_AI_Max_Rate'size use 3 * Word_Size;

for IOS_AI_Max_Rate use
 record
   TBD_Field_One   at 0                    range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for IOS_AI_Max_Rate'size use 3 * Word_Size;

for Navigation_AI_Max_Rate use
 record
   Baro_Out             at  0                    range 0..Word_Size-1;
   Instrument_Heading   at  1 * Word_Size/Bytes range 0..Word_Size-1;
   Heading_Set_Knob     at  2 * Word_Size/Bytes range 0..Word_Size-1;
   Course_Set_Knob      at  3 * Word_Size/Bytes range 0..Word_Size-1;
   ADI_Pitch_Trim       at  4 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for Navigation_AI_Max_Rate'size use 5 * Word_Size;

for Physical_Cues_AI_Max_Rate use
 record
   TBD_Field_One   at 0                    range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for Physical_Cues_AI_Max_Rate'size use 3 * Word_Size;

for Propulsion_AI_Max_Rate use
 record
   Throttle_Position at 0 range 0..Word_Size-1;
 end record;
for Propulsion_AI_Max_Rate'size use 1 * Word_Size;

for Radar_AI_Max_Rate use
```

```
 record
   TBD_Field_One   at 0                    range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for Radar_AI_Max_Rate'size use 3 * Word_Size;

for Visual_AI_Max_Rate use
 record
   TBD_Field_One   at 0                    range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for Visual_AI_Max_Rate'size use 3 * Word_Size;

for Weapons_AI_Max_Rate use
 record
   TBD_Field_One   at 0                    range 0..Word_Size-1;
   TBD_Field_Two   at 1 * Word_Size/Bytes range 0..Word_Size-1;
   TBD_Field_Three at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
for Weapons_AI_Max_Rate'size use 3 * Word_Size;

for Environment_DI_And_State use
 record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Environment_DI_And_State'size use 2 * Bytes;

for Environment_DI_And_State_Array'size use
   Number_Of_Environment_DIs * 2 * Bytes;

for Environment_Discrete_Input_List use
 record
   Number_Of_DIs   at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Environment_DIs * 2 * Bytes)-1;
 end record;
for Environment_Discrete_Input_List'size use
   (Number_Of_Environment_DIs * 2 + 4) * Bytes;
```

```
for Electronic_Warfare_DI_And_State use
 record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Electronic_Warfare_DI_And_State'size use 2 * Bytes;

for Electronic_Warfare_DI_And_State_Array'size use
   Number_Of_Electronic_Warfare_DIs * 2 * Bytes;

for Electronic_Warfare_Discrete_Input_List use
 record
   Number_Of_DIs   at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Electronic_Warfare_DIs * 2 * Bytes)-1;
 end record;
for Electronic_Warfare_Discrete_Input_List'size use
   (Number_Of_Electronic_Warfare_DIs * 2 + 4) * Bytes;

for Flight_Controls_DI_And_State use
 record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Flight_Controls_DI_And_State'size use 2 * Bytes;

for Flight_Controls_DI_And_State_Array'size use
   Number_Of_Flight_Controls_DIs * 2 * Bytes;

for Flight_Controls_Discrete_Input_List use
record
   Number_Of_DIs   at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Flight_Controls_DIs * 2 * Bytes)-1;
end record;
for Flight_Controls_Discrete_Input_List'size use
   (Number_Of_Flight_Controls_DIs * 2 + 4) * Bytes;

for Flight_Dynamics_DI_And_State use
 record
```

```
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Flight_Dynamics_DI_And_State'size use 2 * Bytes;

for Flight_Dynamics_DI_And_State_Array'size use
   Number_Of_Flight_Dynamics_DIs * 2 * Bytes;

for Flight_Dynamics_Discrete_Input_List use
 record
   Number_Of_DIs   at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Flight_Dynamics_DIs * 2 * Bytes)-1;
 end record;
for Flight_Dynamics_Discrete_Input_List'size use
   (Number_Of_Flight_Dynamics_DIs * 2 + 4) * Bytes;

for Flight_Station_DI_And_State use
record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Flight_Station_DI_And_State'size use 2 * Bytes;

for Flight_Station_DI_And_State_Array'size use
   Number_Of_Flight_Station_DIs * 2 * Bytes;

for Flight_Station_Discrete_Input_List use
record
   Number_Of_DIs   at 0 range 0..31;
   Discrete_Inputs at 4 range 0..(38 * Bytes)-1;
 end record;
for Flight_Station_Discrete_Input_List'size use 42 * Bytes;

for IOS_DI_And_State use
 record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for IOS_DI_And_State'size use 2 * Bytes;
```

```
for IOS_DI_And_State_Array'size use
   Number_Of_IOS_DIs * 2 * Bytes;

for IOS_Discrete_Input_List use
 record
   Number_Of_DIs   at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_IOS_DIs * 2 * Bytes)-1;
 end record;
for IOS_Discrete_Input_List'size use
   (Number_Of_IOS_DIs * 2 + 4) * Bytes;

for Navigation_DI_And_State use
record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
end record;
for Navigation_DI_And_State'size use 2 * Bytes;

for Navigation_DI_And_State_Array'size use
   Number_Of_Navigation_DIs * 2 * Bytes;

for Navigation_Discrete_Input_List use
record
   Number_Of_DIs   at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Navigation_DIs * 2 * Bytes)-1;
end record;
for Navigation_Discrete_Input_List'size use
   (Number_Of_Navigation_DIs * 2 + 4) * Bytes;

for Physical_Cues_DI_And_State use
 record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Physical_Cues_DI_And_State'size use 2 * Bytes;

for Physical_Cues_DI_And_State_Array'size use
   Number_Of_Physical_Cues_DIs * 2 * Bytes;
```

```
for Physical_Cues_Discrete_Input_List use
 record
   Number_Of_DIs    at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Physical_Cues_DIs * 2 * Bytes)-1;
 end record;
for Physical_Cues_Discrete_Input_List'size use
   (Number_Of_Physical_Cues_DIs * 2 + 4) * Bytes;

for Propulsion_DI_And_State use
record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
end record;
for Propulsion_DI_And_State'size use 2 * Bytes;

for Propulsion_DI_And_State_Array'size use
   Number_Of_Propulsion_DIs * 2 * Bytes;

for Propulsion_Discrete_Input_List use
record
   Number_Of_DIs    at 0 range 0..Word_Size-1;
   Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
      Number_Of_Propulsion_DIs * 2 * Bytes)-1;
end record;
for Propulsion_Discrete_Input_List'size use
   (Number_Of_Propulsion_DIs * 2 + 4) * Bytes;

for Radar_DI_And_State use
 record
   Name  at 0 range 0..Byte_Size-1;
   State at 1 range 0..Byte_Size-1;
 end record;
for Radar_DI_And_State'size use 2 * Bytes;

for Radar_DI_And_State_Array'size use
   Number_Of_Radar_DIs * 2 * Bytes;

for Radar_Discrete_Input_List use
 record
   Number_Of_DIs    at 0 range 0..Word_Size-1;
```

```
    Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
       Number_Of_Radar_DIs * 2 * Bytes)-1;
  end record;
for Radar_Discrete_Input_List'size use
    (Number_Of_Radar_DIs * 2 + 4) * Bytes;

for Visual_DI_And_State use
  record
    Name  at 0 range 0..Byte_Size-1;
    State at 1 range 0..Byte_Size-1;
  end record;
for Visual_DI_And_State'size use 2 * Bytes;

for Visual_DI_And_State_Array'size use
    Number_Of_Visual_DIs * 2 * Bytes;

for Visual_Discrete_Input_List use
  record
    Number_Of_DIs   at 0 range 0..Word_Size-1;
    Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
       Number_Of_Visual_DIs * 2 * Bytes)-1;
  end record;
for Visual_Discrete_Input_List'size use
    (Number_Of_Visual_DIs * 2 + 4) * Bytes;

for Weapons_DI_And_State use
  record
    Name  at 0 range 0..Byte_Size-1;
    State at 1 range 0..Byte_Size-1;
  end record;
for Weapons_DI_And_State'size use 2 * Bytes;

for Weapons_DI_And_State_Array'size use
    Number_Of_Weapons_DIs * 2 * Bytes;

for Weapons_Discrete_Input_List u
record
    Number_Of_DIs   at 0 range 0..Word_Size-1;
    Discrete_Inputs at 1 * Word_Size/Bytes range 0..(
       Number_Of_Weapons_DIs * 2 * Bytes)-1;
end record;
```

```
for Weapons_Discrete_Input_List'size use
   (Number_Of_Weapons_DIs * 2 + 4) * Bytes;

Stores_Station_Size : constant :=
   Global_Message_Types.Stores_Station_Size;
Weapon_Profile_Size : constant :=
   Global_Message_Types.Weapon_Profile_Size;
Release_Option_Size : constant :=
   Global_Message_Types.Release_Option_Size;
Arming_Option_Size : constant :=
   Global_Message_Types.Arming_Option_Size;
Fuze_Arming_Size : constant :=
   Global_Message_Types.Fuze_Arming_Size;

for Weapon_Station_Option use
 record
   Station
      at  0
      range 0..Stores_Station_Size-1;
   Profile
      at Stores_Station_Size/Bytes
      range 0..Weapon_Profile_Size-1;
   Release
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes
      range 0..Release_Option_Size-1;
   Release_Pulses
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes
      range 0..Byte_Size-1;
   Interval
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes +
         Byte_Size/Bytes
      range 0..Word_Size-1;
   Arming
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes +
```

```
         Byte_Size/Bytes +
         Word_Size/Bytes
      range 0..Arming_Option_Size-1;
   Fuze
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes +
         Byte_Size/Bytes +
         Word_Size/Bytes +
         Arming_Option_Size/Bytes
      range 0..Fuze_Arming_Size-1;
   Arming_Delay1
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes +
         Byte_Size/Bytes +
         Word_Size/Bytes +
         Arming_Option_Size/Bytes +
         Fuze_Arming_Size/Bytes +
         Halfword_Size/Bytes
      range 0..Word_Size-1;
   Arming_Delay2
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes +
         Byte_Size/Bytes +
         Word_Size/Bytes +
         Arming_Option_Size/Bytes +
         Fuze_Arming_Size/Bytes +
         Halfword_Size/Bytes +
         Word_Size/Bytes
      range 0..Word_Size-1;
   Burst_Altitude
      at Stores_Station_Size/Bytes +
         Weapon_Profile_Size/Bytes +
         Release_Option_Size/Bytes +
         Byte_Size/Bytes +
         Word_Size/Bytes +
         Arming_Option_Size/Bytes +
         Fuze_Arming_Size/Bytes +
         Halfword_Size/Bytes +
```

```
            Word_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
      Pull_Up_Range
         at Stores_Station_Size/Bytes +
            Weapon_Profile_Size/Bytes +
            Release_Option_Size/Bytes +
            Byte_Size/Bytes +
            Word_Size/Bytes +
            Arming_Option_Size/Bytes +
            Fuze_Arming_Size/Bytes +
            Halfword_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
      Time_Of_Fall
         at Stores_Station_Size/Bytes +
            Weapon_Profile_Size/Bytes +
            Release_Option_Size/Bytes +
            Byte_Size/Bytes +
            Word_Size/Bytes +
            Arming_Option_Size/Bytes +
            Fuze_Arming_Size/Bytes +
            Halfword_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
   end record;
  for Weapon_Station_Option'size use
      Stores_Station_Size +
      Weapon_Profile_Size +
      Release_Option_Size +
      Byte_Size +
      Word_Size +
      Arming_Option_Size +
      Fuze_Arming_Size +
      Halfword_Size +
      Word_Size +
```

```
    Word_Size +
    Word_Size +
    Word_Size +
    Word_Size;

Jettison_Type_Size : constant :=
    Global_Message_Types.Jettison_Type_Size;

for Jettison_Station use
 record
    Jettison_Kind at 0
                  range 0..Jettison_Type_Size-1;
    Station       at Jettison_Type_Size/Bytes
                  range 0..Stores_Station_Size-1;
 end record;
for Jettison_Station'size use
    Jettison_Type_Size + Stores_Station_Size;

INS_Waypoints_Size : constant :=
    Global_Message_Types.INS_Waypoints_Size;

for UFC_Steerpoint_Selection use
 record
    Steerpoint_Number at 0 range 0..INS_Waypoints_Size-1;
 end record;
for UFC_Steerpoint_Selection'size use INS_Waypoints_Size;

for Speedbrake_Position'size use 8;
for Parking_Brake_Position'size use 8;
for Master_Arm_Switch'size use 8;
for JFS_Start_Switch'size use 8;
for EEC_BUC_Switch'size use 8;
for Starting_Fuel_Switch'size use 8;
for Ralt_Power'size use 8;
for INS_Mode'size use 8;
for Instrument_Mode'size use 8;
for Altimeter_Mode'size use 8;

TACAN_Function_Size : constant := 8;
for TACAN_Function'size use TACAN_Function_Size;
TACAN_Band_Size : constant := 8;
```

```
  for TACAN_Band'size use TACAN_Band_Size;

  for TACAN_Data use
   record
     Channel at 0 range 0..Halfword_Size-1;
     Fun     at Halfword_Size/Bytes range 0..TACAN_Function_Size-1;
     Band    at Halfword_Size/Bytes +
                TACAN_Function_Size/Bytes
             range 0..TACAN_Band_Size-1;
   end record;
  for TACAN_Data'size use
     Halfword_Size +
     TACAN_Function_Size +
     TACAN_Band_Size;

--*
end Flight_Station_Output_Interface_Types;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Flight_Station_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   12 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Flight_Station segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
--  of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
with Flight_Station_Output_Interface_Types;
with Control_Types;
```

```
with Global_Message_Types;
--
package Flight_Station_Output_Interface is
--
--**************************************************************
--*                                                            *
--/ 10.3   Flight Station Output Interface                     *
--*                                                            *
--**************************************************************


--***************Function:
--*
--/ 10.3.1   Electrical System
--*

Electrical_System_Sixteenth_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Electrical_System_Sixteenth_Rate;
--
-- Destination: NAV,IOS,FD,EW,WPN,RDR,PRO,FC,VIS
--

Electrical_System_Quarter_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Electrical_System_Quarter_Rate;
--
-- Destination: PRO
--


--***************Function:
--*
--/ 10.3.2   Hydraulic System
--*

Hydraulic_System_Sixteenth_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Hydraulic_System_Sixteenth_Rate;
--
-- Destination: IOS,FC
```

```
--

Hydraulic_System_Quarter_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Hydraulic_System_Quarter_Rate;
--
-- Destination: NAV,IOS,FD,EW,WPN,RDR,PRO,FC,VIS
--


--***************Function:
--*
--/ 10.3.3   Fuel Management System
--*

Fuel_Management_System_Sixteenth_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Fuel_Management_System_Sixteenth_Rate;
--
-- Destination: IOS,FD
--

Fuel_Management_System_Eighth_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Fuel_Management_System_Eighth_Rate;
--
-- Destination: PRO
--


--***************Function:
--*
--/ 10.3.4   Pneumatic System
--*

Pneumatic_System_Sixteenth_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Pneumatic_System_Sixteenth_Rate;
--
-- Destination: PRO,IOS,PHC,FC
--
```

```
--****************Function:
--*
--/ 10.3.5   Autochecklist System
--*

--NONE


--****************Function:
--*
--/ 10.3.6   Oxygen System
--*

Oxygen_System_Sixteenth_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Oxygen_System_Sixteenth_Rate;
--
-- Destination: IOS
--


--****************Function:
--*
--/ 10.3.7   Crew Station Interface
--*

Crew_Station_Interface_Half_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Crew_Station_Interface_Half_Rate;
--
-- Destination: NAV,IOS,FD,WPN,RDR,PRO,EW,VIS,PHC,FC
--

Electronic_Warfare_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Electronic_Warfare_AI_Max_Rate;
--
-- Destination:  EW,IOS
--
```

```
Flight_Controls_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Flight_Controls_AI_Max_Rate;
--
-- Destination: NAV,IOS,FD,WPN,RDR,PRO,EW,VIS,PHC,FC
--

Flight_Dynamics_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Flight_Dynamics_AI_Max_Rate;
--
-- Destination:  FD,IOS
--

IOS_AI_Max_Fa :e_Outputs :
   Flight_Station_Output_Interface_Types.
   IOS_AI_Max_Rate;
--
-- Destination:  IOS
--

Navigation_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Navigation_AI_Max_Rate;
--
-- Destination:  NAV,IOS
--

Physical_Cues_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Physical_Cues_AI_Max_Rate;
--
-- Destination:  PC,IOS
--

Propulsion_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Propulsion_AI_Max_Rate;
--
-- Destination:  PRO,IOS
--
```

```
Radar_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Radar_AI_Max_Rate;
--
-- Destination:  RDR,IOS
--

Visual_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Visual_AI_Max_Rate;
--
-- Destination:  VIS,IOS
--

Weapons_AI_Max_Rate_Outputs :
   Flight_Station_Output_Interface_Types.
   Weapons_AI_Max_Rate;
--
-- Destination:  WPN,IOS
--


--SEND-ON-CHANGE OUTPUTS

Electronic_Warfare_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Electronic_Warfare_Discrete_Input_List;
--
-- Destination:  EW,IOS
--

Flight_Controls_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Flight_Controls_Discrete_Input_List;
--
-- Destination:  FC,IOS
--

Flight_Dynamics_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Flight_Dynamics_Discrete_Input_List;
```

```
--
-- Destination:  FD,IOS
--

Flight_Station_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Flight_Station_Discrete_Input_List;
--
-- Destination: NAV,WPN,IOS
--

IOS_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   IOS_Discrete_Input_List;
--
-- Destination:  IOS
--

Navigation_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Navigation_Discrete_Input_List;
--
-- Destination:  NAV
--

Physical_Cues_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Physical_Cues_Discrete_Input_List;
--
-- Destination:  PHC,IOS
--

Propulsion_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Propulsion_Discrete_Input_List;
--
-- Destination:  PRO,IOS
--

Radar_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
```

```
   Radar_Discrete_Input_List;
--
-- Destination:  RDR,IOS
--

Visual_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Visual_Discrete_Input_List;
--
-- Destination:  VIS,IOS
--

Weapons_Discrete_Input_Change :
   Flight_Station_Output_Interface_Types.
   Weapons_Discrete_Input_List;
--
-- Destination:  WPN
--

MFD_Mode_Outputs :
   Global_Message_Types.
   Master_Mode;
--
-- Destination: NAV,WPN
--

SMS_Mode_Outputs :
   Global_Message_Types.
   SMS_Submode;
--
-- Destination: WPN
--

A_G_WPN_Mode_Outputs :
   Global_Message_Types.
   A_G_Weapon_Delivery_Mode;
--
-- Destination: WPN
--

Stores_Configuration :
```

```
   Global_Message_Types.
   Weapon_Station_Change;
--
-- Destination: WPN, IOS
--

Current_Station_Change_Of_State :
   Global_Message_Types.
   Stores_Station;
--
-- Destination: WPN
--

Weapon_Option_Outputs :
   Flight_Station_Output_Interface_Types.
   Weapon_Station_Option;
--
-- Destination: WPN
--

TACAN_Data_Change_Of_State :
   Flight_Station_Output_Interface_Types.
   TACAN_Data;
--
-- Destination: NAV
--

Current_Jett_Station_Change_Of_State :
   Flight_Station_Output_Interface_Types.
   Jettison_Station;
--
-- Destination: WPN
--

UFC_Outputs :
   Flight_Station_Output_Interface_Types.
   UFC_Steerpoint_Selection;
--
-- Destination: NAV
--
```

Switch_Change_Of_State :
t‹· 'on_Output_Interface_Types.
: ›sition;

ion: ENV, FC, IOS

‹e_Switch_Change_Of_State :
:ation_Output_Interface_Types.
3rake_Position;

ion: FC,IOS

3witch_Change_Of_State :
:ation_Output_Interface_Types.
:m_Switch;

.on: WPN

_Change_Of_State :
:ation_Output_Interface_Types.
_Switch;

on: PRO

ch_Change_Of_State :
ation_Output_Interface_Types.
witch;

on: PRO

l_Switch_Change_Of_State :
ation_Output_Interface_Types.
Fuel_Switch;

PRO

```
    .on_Segment_Training_Mode_Response :
rol_Types.
ent_Training_Mode_Response;

ination : IOS


Station_Performance_Test_Response :
rol_Types.
ormance_Test_Response;

ination : IOS


Station_Off_Line_Diagnostic_Response :
rol_Types.
Line_Diagnostic_Response;

ir  ion : IOS


Station_Remote_Controlled_Diagnostic_Response :
rol_Types.
te_Controlled_Diagnostic_Response;

ination : IOS


Station_On_Line_Diagnostic_Response :
rol_Types.
ine_Diagnostic_F .ponse;

ination : IOS


Station_Scoring_Response :
rol_Types.
  Response;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Flight_Controls_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Flight_Controls segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Flight_Controls_Output_Interface_Types is
--
--*************************************************************
--*                                                           *
--/ 10.4   Flight Controls Output Interface Types             *
--*                                                           *
--*************************************************************

--*****************************************
```

```
--*
--/ 10.4.1   Aircraft/Simulator Specific Flight Controls Types
--*
--******************************************

-- Declare all D/Os in the flight station that this segment will
-- turn on or off.
type Flight_Controls_Discrete_Outputs is (
   Flt_Control_System_Caution_Light,
   LE_Flaps_Caution_Light,
   Anti_Skid_Caution_Light,
   Hook_Caution_Light,
   NWS_Fail_Caution_Light,
   Stick_Override_Light,
   Landing_Gear_Handle_Light,
   Dual_FC_Fail_Warning_Light,
   Landing_Gear_Warning_Horn,
   Takeoff_Landing_Config_Warning_Light,
   Left_Horizontal_Tail_Servo_Status_Light,
   Right_Horizontal_Tail_Servo_Status_Light,
   Left_Flaperon_Servo_Status_Light,
   Right_Flaperon_Servo_Status_Light,
   Rudder_Servo_Status_Light,
   Right_Standby_Gains_Light,
   Pitch_Status_Light,
   Yaw_Status_Light,
   Roll_Status_Light,
   FLCC_Data_Word_Dot_Light,
   ECA_Data_Word_Dot_Light,
   Rate_Gyro_Speed_Detect_Dot_Light,
   Test_Adv_Mal_Light,
   Left_Wheel_Down_Light,
   Right_Wheel_Down_Light,
   Nose_Wheel_Down_Light,
   Speedbrakes_Closed_Indicator,
   Speedbrakes_Open_Indicator);

-- Axes which may be trimmed in this aircraft
type Aircraft_Trim is (
   Heading,
   Pitch,
```

```
   Roll);


--***************************************
--*
--/ 10.4.2   Aircraft/Simulator Reusable Flight Controls Types
--*
--***************************************

-- Must declare this size here because Flight_Controls_Discrete_Outputs
-- is used as an index, which forces its representation
Flight_Controls_Discrete_Outputs_Size : constant := 8;
for Flight_Controls_Discrete_Outputs'size use
   Flight_Controls_Discrete_Outputs_Size;

Number_Of_Flight_Controls_Discrete_Outputs : constant :=
   Flight_Controls_Discrete_Outputs'pos (
   Flight_Controls_Discrete_Outputs'last) -
   Flight_Controls_Discrete_Outputs'pos (
   Flight_Controls_Discrete_Outputs'first) + 1;

-- Landing gear parameters
type Landing_Gear_State is (
   Locked_Up,
   Up,
   Retracting,
   Extending,
   Down,
   Locked_Down);

-- Some segments want landing gear position, while others want
-- landing gear state.  We include both.
type Landing_Gear_Parameters is
 record
   Position    : Engineering_Units.Normalized;
                 --0.0=fully retracted,
                 --1.0=fully extended
   State       : Landing_Gear_State;
   Crab_Angle  : Engineering_Units.Signed_Degrees;
                 --0.0=no crab angle;
                 -- positive (negative) angle indicates gear is pointed
```

```
                                  -- right (left) of x-body axis
 end record;

-- Adapt the declaration of Aircraft_Landing_Gear
-- in Global_Message_Types
type Aircraft_Landing_Gear_Status_Array is
   array (Global_Message_Types.Aircraft_Landing_Gear) of
   Landing_Gear_Parameters;

-- Doors and hatches
type Door_And_Hatch_State is (
   Locked_Open,
   Open,
   Opening,
   Closing,
   Closed,
   Locked_Closed);

type Door_Or_Hatch_Data is
 record
   Position : Engineering_Units.Normalized;
              --0.0=fully closed
              --1.0=fully open
   State    : Door_And_Hatch_State;
 end record;

-- Adapt the declaration of Doors_And_Hatches
-- in Global_Message_Types
type Aircraft_Door_Status_Array is
   array (Global_Message_Types.Aircraft_Doors_And_Hatches) of
   Door_Or_Hatch_Data;

-- Brake pressure
type Wheel_Brake_Pressure_Array is
  array (Global_Message_Types.Aircraft_Wheel) of Engineering_Units.PSI;

-- Secondary control surfaces
-- Adapt the declaration of Secondary_Control_Surface
-- in Global_Message_Types
type Secondary_Control_Surface_Position_Array is
   array (Global_Message_Types.Secondary_Control_Surface) of
```

```
   Engineering_Units.Signed_Normalized;
   --For cases where the surface moves in (forward) and
   --out (aft), 0.0=fully in, forward, or retracted,
      --1.0=fully out, aft, or extended.
      --For cases where the surface moves up (right) and
   --down (left) relative to a streamline position, the
   --normalization is based upon the maximum displacement
   --from the streamline position.
   --In the case of symmetrical displacement from the
   --streamline, -1.0=down (left), 0.0=null, and 1.0=up
   --(right).

   -- Aircraft specifics can be added here:
   -- Leading Edge Flaps: 0.0 = ? deg; 1.0 = ? deg
   -- Speedbrakes: 0.0 = ? deg; 1.0 = ? deg
   -- Nosewheel: -1.0 = ? deg; 1.0 = ? deg

-- Primary control surfaces
-- Adapt the declaration of Primary_Control_Surface
-- in Global_Message_Types
type Primary_Control_Surface_Deflection_Array is
   array (Global_Message_Types.Primary_Control_Surface) of
   Engineering_Units.Signed_Degrees;

   --Positive angle indicates up (right) deflection;
   --Negative angle indicates down (left) deflection;
   --0.0 degrees indicates the null position.

-- Cockpit control devices
-- Adapt the declaration of Cockpit_Control_Device
-- in Global_Message_Types
-- Message_Types for this aircraft
type Cockpit_Control_Device_Position_Array is
    array (Global_Message_Types.Cockpit_Control_Device) of
    Engineering_Units.Signed_Normalized;
   --For cases where the control device moves forward
   --(right) and aft (left) relative to a null position,
   -- -1.0=aft (left), 0.0=null, and 1.0=forward (right).
   --For cases where the control device moves unilaterally
   --from a null position to a maximum position, 0.0=null
   --and 1.0=maximum.
```

```
-- Unique control devices
-- Adapt the declaration of Aircraft_Unique_Control_Device
-- in Global_Message_Types
type Unique_Control_Device_Position_Array is
   array (Global_Message_Types.Aircraft_Unique_Control_Device) of
   Engineering_Units.Signed_Normalized;
   --For cases where the control device moves forward
   --(right) and aft (left) relative to a null position,
   -- -1.0=aft (left), 0.0=null, and 1.0=forward (right).
   --For cases where the control device moves unilaterally
   --from a null position to a maximum position, 0.0=null
   --and 1.0=maximum.

-- Trim
type Aircraft_Trim_Position_Array is
   array (Aircraft_Trim) of Engineering_Units.Signed_Normalized;
   --For cases where the surface moves up (right) and
   --down (left) relative to a null position, the
   --normalization is based upon the maximum displacement
   --from the null position.
   --In the case of symmetrical displacement from the
   --null, -1.0=down (left), 0.0=null, and 1.0=up (right).

-- Surface tabs
-- Adapt the declaration of Aircraft_Surface_Tab
-- in Global_Message_Types
type Surface_Tab_Deflection_Array is
   array (Global_Message_Types.Aircraft_Surface_Tab) of
   Engineering_Units.Signed_Degrees;
   --Positive angle indicates up (right) deflection.
   --Negative angle indicates down (left) deflection.
   --0.0 degrees indicates the null position.

-- Hydraulic components
-- Adapt the declaration of Flight_Controls_Hydraulic_Component
-- in Global_Message_Types
type Flight_Controls_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types.Flight_Controls_Hydraulic_Component)
   of Engineering_Units.Gal_Per_Min;
```

```
-- Pneumatic components
-- Adapt the declaration of Flight_Controls_Pneumatic_Compc- ent
-- in Global_Message_Types for this aircraft
type Flight_Controls_Pneumatic_Component_Flow_Array is
   array (Global_Message_Types.Flight_Controls_Pneumatic_Component)
   of Engineering_Units.Ft3_Per_Min;

-- Throttles
-- Adapt the declaration of Aircraft_Throttle_Lever
-- in Global_Message_Types
type Throttle_Array is
   array (Global_Message_Types.Aircraft_Throttle_Lever) of
   Engineering_Units.Normalized;
      --0.0=minimum throttle,
      --1.0=maximum throttle


--*****************************************
--*
--/ 10.4.3   Flight Controls Segment Output Records
--*
--*****************************************


--***************Function:
--*
--/ 10.4.3.1   Primary Controls
--*

type Primary_Controls_Max_Rate is
 record
   Surface_Deflection       : Primary_Control_Surface_Deflection_Array;
                              --FL, FS, IOS, PHC
   Cockpit_Control_Position : Cockpit_Control_Device_Position_Array;
                              --IOS
 end record;


--***************Function:
--*
--/ 10.4.3.2   Misc Control Devices
```

```
--*

type Misc_Control_Devices_Quarter_Rate is
 record
   Secondary_Control_Surface_Position :
Secondary_Control_Surface_Position_Array;
                                          --FD, FS, IOS, PHC
   Aircraft_Landing_Gear_Status      : Aircraft_Landing_Gear_Status_Array;
                                          --FD, FS, IOS, PHC
   Aircraft_Door_Status                 : Aircraft_Door_Status_Array;
                                          --FD, FS, PHC
   Unique_Control_Device_Positions   : Unique_Control_Device_Position_Array;
                                          --FD, FS
 end record;

-- SEND-ON-CHANGE

type Arresting_Hook_State is (
   Up,
   Down);


--***************Function:
--*
--/ 10.4.3.3   Trim
--*

type Trim_Max_Rate is
 record
   Aircraft_Trim_Positions : Aircraft_Trim_Position_Array; --FS
   Surface_Tab_Deflections : Surface_Tab_Deflection_Array; --FD
 end record;


--***************Function:
--*
--/ 10.4.3.4   Toe Brakes and Anti-Skid
--*

type Toe_Brakes_And_Anti_Skid_Quarter_Rate is
 record
   Wheel_Brake_Pressure : Wheel_Brake_Pressure_Array; --FD, PHC
```

```
 end record;


--***************Function:
--*
--/ 10.4.3.5   AFCS
--*

type AFCS_Quarter_Rate is
 record
   Commanded_Throttle_Position : Throttle_Array;   --FS
 end record;


--***************Function:
--*
--/ 10.4.3.6   Hinge Moments
--*

--NONE


--***************Function:
--*
--/ 10.4.3.7   Flight Controls Support
--*

-- ITERATIVE

type Flight_Controls_Support_Eighth_Rate is
 record
   Electrical_Loads           : Global_Message_Types.
                                Aircraft_Electrical_Bus_Load_Array; --FS
  Hydraulic_Component_Flows : Flight_Controls_Hydraulic_Component_Flow_Array;
                                --FS
  Pneumatic_Component_Flows : Flight_Controls_Pneumatic_Component_Flow_Array;
                                --FS
 end record;

-- SEND-ON-CHANGE
```

```
-- See Control_Types for responses to IOS

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Flight_Controls_Discrete_Output_And_State is
record
   Name  : Flight_Controls_Discrete_Outputs;
   State : Base_Types.Discrete_State;
 end record;

--
-- ...or it will collect them into an array;
--
subtype Flight_Controls_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32
   range 1..Number_Of_Flight_Controls_Discrete_Outputs;

type Flight_Controls_Discrete_Output_Array is array (
   Flight_Controls_Discrete_Output_Count) of
   Flight_Controls_Discrete_Output_And_State;

--
-- ...and send the ones which have changed in one of these messages:
--
type Flight_Controls_Discrete_Output_List is
 record
   Number_Of_DOs     : Flight_Controls_Discrete_Output_Count;
   Discrete_Outputs  : Flight_Controls_Discrete_Output_Array;
 end record;


--*******************************************
--*
--/ 10.4.4   Flight Controls Representation Specs
--*
--*******************************************
private

   --
```

```
-- Declarations to make representation specs more readable
--
Bytes     : constant := 8; -- Bits per byte
Byte_Size : constant := 1 * Bytes;
Word_Size : constant := 4 * Bytes;

-- 10.4.3.1
Number_Of_Primary_Control_Surfaces : constant :=
   Global_Message_Types.Primary_Control_Surface'pos (
   Global_Message_Types.Primary_Control_Surface'last) -
   Global_Message_Types.Primary_Control_Surface'pos (
   Global_Message_Types.Primary_Control_Surface'first) + 1;

Primary_Control_Surface_Deflection_Array_Size : constant :=
   Number_Of_Primary_Control_Surfaces * Word_Size;
for Primary_Control_Surface_Deflection_Array'size use
   Primary_Control_Surface_Deflection_Array_Size;

Number_Of_Cockpit_Control_Devices : constant :=
   Global_Message_Types.Cockpit_Control_Device'pos (
   Global_Message_Types.Cockpit_Control_Device'last) -
   Global_Message_Types.Cockpit_Control_Device'pos (
   Global_Message_Types.Cockpit_Control_Device'first) + 1;

Cockpit_Control_Device_Position_Array_Size : constant :=
   Number_Of_Cockpit_Control_Devices * Word_Size;
for Cockpit_Control_Device_Position_Array'size use
   Cockpit_Control_Device_Position_Array_Size;

for Primary_Controls_Max_Rate use
 record
   Surface_Deflection       at 0
      range 0..Primary_Control_Surface_Deflection_Array_Size-1;
   Cockpit_Control_Position at
      Primary_Control_Surface_Deflection_Array_Size/Bytes
      range 0..Cockpit_Control_Device_Position_Array_Size-1;
 end record;
for Primary_Controls_Max_Rate'size use
   Primary_Control_Surface_Deflection_Array_Size +
   Cockpit_Control_Device_Position_Array_Size;
```

```
-- 10.4.3.2
Number_Of_Secondary_Control_Surfaces : constant :=
   Global_Message_Types.Secondary_Control_Surface'pos (
   Global_Message_Types.Secondary_Control_Surface'last) -
   Global_Message_Types.Secondary_Control_Surface'pos (
   Global_Message_Types.Secondary_Control_Surface'first) + 1;

Secondary_Control_Surface_Position_Array_Size : constant :=
   Number_Of_Secondary_Control_Surfaces * Word_Size;
for Secondary_Control_Surface_Position_Array'size use
   Secondary_Control_Surface_Position_Array_Size;

for Landing_Gear_State'size use Byte_Size;

-- Note that pad bytes are declared in this record, so that the
-- record can be an element in an array without causing
-- alignment problems.
for Landing_Gear_Parameters use
 record
   Position   at 0                     range 0..Word_Size-1;
   State      at 1 * Word_Size/Bytes range 0..Byte_Size-1;
   -- 3 bytes spare
   Crab_Angle at 2 * Word_Size/Bytes range 0..Word_Size-1;
 end record;
Landing_Gear_Parameters_Size : constant := 3 * Word_Size;
for Landing_Gear_Parameters'size use Landing_Gear_Parameters_Size;

Number_Of_Landing_Gear : constant :=
   Global_Message_Types.Aircraft_Landing_Gear'pos (
   Global_Message_Types.Aircraft_Landing_Gear'last) -
   Global_Message_Types.Aircraft_Landing_Gear'pos (
   Global_Message_Types.Aircraft_Landing_Gear'first) + 1;

Aircraft_Landing_Gear_Status_Array_Size : constant :=
   Number_Of_Landing_Gear * Landing_Gear_Parameters_Size;
for Aircraft_Landing_Gear_Status_Array'size use
   Aircraft_Landing_Gear_Status_Array_Size;

for Door_And_Hatch_State'size use Byte_Size;
--
-- Note that pad bytes are declared in this record, so that the
```

```
-- record can be an element in an array without causing
-- alignment problems.
--
for Door_Or_Hatch_Data use
 record
   Position at 0               range 0..Word_Size-1;
   State    at Word_Size/Bytes range 0..Byte_Size-1;
   -- 3 bytes spare
 end record;
Door_Or_Hatch_Data_Size : constant := 2 * Word_Size;
for Door_Or_Hatch_Data'size use Door_Or_Hatch_Data_Size;

Number_Of_Doors_And_Hatches : constant :=
   Global_Message_Types.Aircraft_Doors_And_Hatches'pos (
   Global_Message_Types.Aircraft_Doors_And_Hatches'last) -
   Global_Message_Types.Aircraft_Doors_And_Hatches'pos (
   Global_Message_Types.Aircraft_Doors_And_Hatches'first) + 1;

Aircraft_Door_Status_Array_Size : constant :=
   Number_Of_Doors_And_Hatches * Door_Or_Hatch_Data_Size;
for Aircraft_Door_Status_Array'size use
   Aircraft_Door_Status_Array_Size;

Number_Of_Aircraft_Unique_Control_Devices : constant :=
   Global_Message_Types.Aircraft_Unique_Control_Device'pos (
   Global_Message_Types.Aircraft_Unique_Control_Device'last) -
   Global_Message_Types.Aircraft_Unique_Control_Device'pos (
   Global_Message_Types.Aircraft_Unique_Control_Device'first) + 1;

Unique_Control_Device_Position_Array_Size : constant :=
   Number_Of_Aircraft_Unique_Control_Devices * Word_Size;
for Unique_Control_Device_Position_Array'size use
   Unique_Control_Device_Position_Array_Size;

for Misc_Control_Devices_Quarter_Rate use
 record
   Secondary_Control_Surface_Position
      at 0
      range 0..Secondary_Control_Surface_Position_Array_Size-1;
   Aircraft_Landing_Gear_Status
      at Secondary_Control_Surface_Position_Array_Size/Bytes
```

```
      range 0..Aircraft_Landing_Gear_Status_Array_Size-1;
   Aircraft_Door_Status
      at Secondary_Control_Surface_Position_Array_Size/Bytes +
         Aircraft_Landing_Gear_Status_Array_Size/Bytes
      range 0..Aircraft_Door_Status_Array_Size-1;
   Unique_Control_Device_Positions
      at Secondary_Control_Surface_Position_Array_Size/Bytes +
         Aircraft_Landing_Gear_Status_Array_Size/Bytes +
         Aircraft_Door_Status_Array_Size/Bytes
      range 0..Unique_Control_Device_Position_Array_Size-1;
 end record;
for Misc_Control_Devices_Quarter_Rate'size use
   Secondary_Control_Surface_Position_Array_Size +
   Aircraft_Landing_Gear_Status_Array_Size +
   Aircraft_Door_Status_Array_Size +
   Unique_Control_Device_Position_Array_Size;

Arresting_Hook_State_Size : constant := Byte_Size;
for Arresting_Hook_State'size use Arresting_Hook_State_Size;

-- 10.4.3.3
Aircraft_Trim_Size : constant := Byte_Size;
for Aircraft_Trim'size use Aircraft_Trim_Size;

Number_Of_Aircraft_Trims : constant :=
   Aircraft_Trim'pos (Aircraft_Trim'last) -
   Aircraft_Trim'pos (Aircraft_Trim'first) + 1;

Aircraft_Trim_Position_Array_Size : constant :=
   Number_Of_Aircraft_Trims * Word_Size;
for Aircraft_Trim_Position_Array'size use
   Aircraft_Trim_Position_Array_Size;

Number_Of_Aircraft_Surface_Tabs : constant :=
   Global_Message_Types.Aircraft_Surface_Tab'pos (
   Global_Message_Types.Aircraft_Surface_Tab'last) -
   Global_Message_Types.Aircraft_Surface_Tab'pos (
   Global_Message_Types.Aircraft_Surface_Tab'first) + 1;

Surface_Tab_Deflection_Array_Size : constant :=
   Number_Of_Aircraft_Surface_Tabs * Word_Size;
```

```
for Surface_Tab_Deflection_Array'size use
   Surface_Tab_Deflection_Array_Size;

for Trim_Max_Rate use
 record
   Aircraft_Trim_Positions
      at 0
      range 0..Aircraft_Trim_Position_Array_Size-1;
   Surface_Tab_Deflections
      at Aircraft_Trim_Position_Array_Size/Bytes
      range 0..Surface_Tab_Deflection_Array_Size-1;
 end record;
for Trim_Max_Rate'size use
   Aircraft_Trim_Position_Array_Size +
   Surface_Tab_Deflection_Array_Size;

-- 10.4.3.4
Number_Of_Aircraft_Wheels : constant :=
   Global_Message_Types.Aircraft_Wheel'pos (
   Global_Message_Types.Aircraft_Wheel'last) -
   Global_Message_Types.Aircraft_Wheel'pos (
   Global_Message_Types.Aircraft_Wheel'first) + 1;

Wheel_Brake_Pressure_Array_Size : constant :=
   Number_Of_Aircraft_Wheels * Word_Size;
for Wheel_Brake_Pressure_Array'size use
   Wheel_Brake_Pressure_Array_Size;

for Toe_Brakes_And_Anti_Skid_Quarter_Rate use
 record
  Wheel_Brake_Pressure at 0 range 0..Wheel_Brake_Pressure_Array_Size-1;
    --FD, PHC
 end record;
for Toe_Brakes_And_Anti_Skid_Quarter_Rate'size use
   Wheel_Brake_Pressure_Array_Size;

-- 10.4.3.5
Number_Of_Aircraft_Throttle_Levers : constant :=
   Global_Message_Types.Aircraft_Throttle_Lever'pos (
   Global_Message_Types.Aircraft_Throttle_Lever'last) -
   Global_Message_Types.Aircraft_Throttle_Lever'pos (
```

```
      Global_Message_Types.Aircraft_Throttle_Lever'first) + 1;

   Throttle_Array_Size : constant :=
      Number_Of_Aircraft_Throttle_Levers * Word_Size;
   for Throttle_Array'size use Throttle_Array_Size;

   for AFCS_Quarter_Rate use
    record
      Commanded_Throttle_Position at 0 range 0..Throttle_Array_Size-1;
    end record;
   for AFCS_Quarter_Rate'size use Throttle_Array_Size;

   -- 10.4.3.7
   Aircraft_Electrical_Bus_Load_Array_Size : constant :=
      Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size;

   Number_Of_Flight_Controls_Hydraulic_Components : constant :=
      Global_Message_Types.Flight_Controls_Hydraulic_Component'pos (
      Global_Message_Types.Flight_Controls_Hydraulic_Component'last) -
      Global_Message_Types.Flight_Controls_Hydraulic_Component'pos (
      Global_Message_Types.Flight_Controls_Hydraulic_Component'first) +
1;

   Flight_Controls_Hydraulic_Component_Flow_Array_Size : constant :=
      Number_Of_Flight_Controls_Hydraulic_Components * Word_Size;
   for Flight_Controls_Hydraulic_Component_Flow_Array'size use
      Flight_Controls_Hydraulic_Component_Flow_Array_Size;

   Number_Of_Flight_Controls_Pneumatic_Components : constant :=
      Global_Message_Types.Flight_Controls_Pneumatic_Component'pos (
      Global_Message_Types.Flight_Controls_Pneumatic_Component'last) -
      Global_Message_Types.Flight_Controls_Pneumatic_Component'pos (
      Global_Message_Types.Flight_Controls_Pneumatic_Component'first) +
1;

   Flight_Controls_Pneumatic_Component_Flow_Array_Size : constant :=
      Number_Of_Flight_Controls_Pneumatic_Components * Word_Size;
   for Flight_Controls_Pneumatic_Component_Flow_Array'size use
      Flight_Controls_Pneumatic_Component_Flow_Array_Size;

   for Flight_Controls_Support_Eighth_Rate use
    record
```

```
   Electrical_Loads
      at 0
      range 0..Aircraft_Electrical_Bus_Load_Array_Size-1;
   Hydraulic_Component_Flows
      at Aircraft_Electrical_Bus_Load_Array_Size/Bytes
     range 0..Flight_Controls_Hydraulic_Component_Flow_Array_Size-1;
   Pneumatic_Component_Flows
      at Aircraft_Electrical_Bus_Load_Array_Size/Bytes +
         Flight_Controls_Hydraulic_Component_Flow_Array_Size/Bytes
     range 0..Flight_Controls_Pneumatic_Component_Flow_Array_Size-1;
 end record;
for Flight_Controls_Support_Eighth_Rate'size use
   Aircraft_Electrical_Bus_Load_Array_Size +
   Flight_Controls_Hydraulic_Component_Flow_Array_Size +
   Flight_Controls_Pneumatic_Component_Flow_Array_Size;

for Flight_Controls_Discrete_Output_And_State use
record
   Name  at 0
         range 0..Flight_Controls_Discrete_Outputs_Size-1;
   State at Flight_Controls_Discrete_Outputs_Size/Bytes
         range 0..Byte_Size-1;
 end record;
Flight_Controls_Discrete_Output_And_State_Size : constant :=
   Flight_Controls_Discrete_Outputs_Size + Byte_Size;
for Flight_Controls_Discrete_Output_And_State'size use
   Flight_Controls_Discrete_Output_And_State_Size;

Flight_Controls_Discrete_Output_Array_Size : constant :=
   Flight_Controls_Discrete_Output_And_State_Size *
   Number_Of_Flight_Controls_Discrete_Outputs;
for Flight_Controls_Discrete_Output_Array'size use
   Flight_Controls_Discrete_Output_Array_Size;

for Flight_Controls_Discrete_Output_List use
record
   Number_Of_DOs    at 0               range 0..Word_Size-1;
   Discrete_Outputs at Word_Size/Bytes range 0..
      Flight_Controls_Discrete_Output_Array_Size-1;
 end record;
for Flight_Controls_Discrete_Output_List'size use
```

```
Flight_Controls_Discrete_Output_Array_Size + Word_Size;

end Flight_Controls_Output_Interface_Types;
```

```
-----------------------------------------------------------------
-- %Z% Unit Name:        Flight_Controls_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   12 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Flight_Controls segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
-- of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
with Flight_Controls_Output_Interface_Types;
with Control_Types;
```

```
--
package Flight_Controls_Output_Interface is
--
--*****************************************************************
--*                                                               *
--/ 10.5   Flight Controls Output Interface                       *
--*                                                               *
--*****************************************************************


--***************Function:
--*
--/ 10.5.1   Primary Controls
--*

Primary_Controls_Max_Rate_Outputs :
   Flight_Controls_Output_Interface_Types.
   Primary_Controls_Max_Rate;
--
-- Destination: ENV, FD, FS, IOS, PHC
--


--***************Function:
--*
--/ 10.5.2   Misc Control Devices
--*

Misc_Control_Devices_Quarter_Rate_Outputs :
   Flight_Controls_Output_Interface_Types.
   Misc_Control_Devices_Quarter_Rate;
--
-- Destination:  ENV, FD, FS, IOS, PHC
--

-- SEND-ON-CHANGE OUTPUTS

Current_Arresting_Hook_State :
   Flight_Controls_Output_Interface_Types.
   Arresting_Hook_State;
--
```

```
-- Destination:  ENV, FD, FS
--

--***************Function:
--*
--/ 10.5.3   Trim
--*

Trim_Max_Rate_Outputs :
   Flight_Controls_Output_Interface_Types.
   Trim_Max_Rate;
--
-- Destination:  FD, FS
--


--***************Function:
--*
--/ 10.5.4   Toe Brakes and Anti-Skid
--*

Toe_Brakes_And_Anti_Skid_Quarter_Rate_Outputs :
   Flight_Controls_Output_Interface_Types.
   Toe_Brakes_And_Anti_Skid_Quarter_Rate;
--
-- Destination:  FD, PHC
--


--***************Function:
--*
--/ 10.5.5   AFCS
--*

AFCS_Quarter_Rate_Outputs :
   Flight_Controls_Output_Interface_Types.
   AFCS_Quarter_Rate;
--
-- Destination:  FS
--
```

```
--***************Function:
--*
--/ 10.5.6   Hinge Moments
--*

--NONE


--***************Function:
--*
--/ 10.5.7   Flight Controls Support
--*

Flight_Controls_Support_Eighth_Rate_Outputs :
   Flight_Controls_Output_Interface_Types.
   Flight_Controls_Support_Eighth_Rate;
--
-- Destination:  FS
--

-- SEND-ON-CHANGE OUTPUTS

Flight_Controls_Discrete_Output_Change :
   Flight_Controls_Output_Interface_Types.
   Flight_Controls_Discrete_Output_List;
--
-- Destination:  FS
--

Flight_Controls_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Flight_Controls_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
```

```
-- Destination : IOS
--

Flight_Controls_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Flight_Controls_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Flight_Controls_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Flight_Controls_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Flight_Controls_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--

--*
end Flight_Controls_Output_Interface;
```

```
----------------------------------------------------------------------
-- %Z% Unit Name:        Flight_Dynamics_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Flight_Dynamics segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Flight_Dynamics_Output_Interface_Types is
--
--*************************************************************
--*                                                           *
--/ 10.6   Flight Dynamics Output Interface Types             *
--*                                                           *
--*************************************************************


--****************************************
```

```
--*
--/ 10.6.1   Aircraft/Simulator Specific Flight Dynamics Types
--*
--*****************************************

type Touchdown_Location is (
   Left_Wheel,
   Right_Wheel,
   Left_Wing,
   Right_Wing,
   Tail,
   Nose);


--*****************************************
--*
--/ 10.6.2   Aircraft/Simulator Reusable Flight Dynamics Types
--*
--*****************************************

type Flight_Dynamics_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types.Flight_Dynamics_Hydraulic_Component)
   of Engineering_Units.Gal_Per_Min;

type Buffet_Status is
 record
   Buffet_Vibration : Global_Message_Types.Vibration_Characteristics;
--PHC
   Buffet_State     : Base_Types.Discrete_State;   --PHC
end record;

type Touchdown_State is (
   In_Air,
   On_Ground,
   On_Sea);

type Wheel_Status_Data is
 record
   Wheel_Speed      : Engineering_Units.RPM;       --PHC, FC
   Tire_Blow_Out    : Base_Types.Discrete_State;  --PHC
   Tire_Slip_Angle  : Engineering_Units.Degrees;  --PHC
   Tire_Skid        : Base_Types.Discrete_State;  --PHC, FC
```

```
end record;

type Landing_Gear_Compression_Rate_Array is
   array (Global_Message_Types.Aircraft_Landing_Gear) of
   Engineering_Units.Feet_Per_Sec;

type Weight_On_Wheels_Array is
   array (Global_Message_Types.Aircraft_Landing_Gear) of
   Base_Types.Discrete_State;

type Wheel_Status_Array is
   array (Global_Message_Types.Aircraft_Wheel) of Wheel_Status_Data;


--****************************************
--*
--/ 10.6.3   Flight Dynamics Segment Output Records
--*
--****************************************


--***************Function:
--*
--/ 10.6.3.1   Equations of Motion
--*

type Equations_Of_Motion_Max_Rate is
 record
   Flight_Parameters_Wind_Axis        : Engineering_Units.
                                        Angular_Position_Components;
                                        --FC, IOS,NAV,PHC, WPN
   Ownship_Angular_Acceleration       : Engineering_Units.
                                        Angular_Acceleration_Components;
                                        --PHC
   Ownship_Angular_Position           : Engineering_Units.
                                        Angular_Position_Components;
                                        --PHC, NAV
   Ownship_Angular_Velocity           : Engineering_Units.
                                        Angular_Velocity_Components;
                                        --IOS, NAV, PRO, VIS, FC
   Ownship_Attitude_Relative_To_Deck  : Engineering_Units.
```

```
                                         Angular_Position_Components;
                                         --NAV
   Ownship_Earth_Axis_Acceleration     : Engineering_Units.
                                         Earth_Acceleration_Components;
                                         --NAV, PHC
   Ownship_Earth_Axis_Position         : Engineering_Units.
                                         Earth_Position_Components;
                                         --EW, IOS, NAV.RDR, VIS
   Ownship_Earth_Axis_Velocity         : Engineering_Units.
                                         Earth_Velocity_Components;
                                         --EW, IOS, NAV, RDR
   Ownship_Linear_Acceleration         : Engineering_Units.
                                         Linear_Acceleration_Components;
                                         --NAV, PHC
   Ownship_Linear_Velocity             : Engineering_Units.
                                         Linear_Velocity_Components;
                                         --NAV
   Buffet_Output                       : Buffet_Status;
                                         --PHC
end record;

type Equations_Of_Motion_Quarter_Rate is
 record
   Equivalent_Airspeed : Engineering_Units.Knots;      --PHC
   Mach_Number         : Engineering_Units.Mach_Range; -- FC, PRO, IOS,
PHC, NAV
   Ground_Speed        : Engineering_Units.Knots;      --PHC, NAV, IOS,
EW
   True_Airspeed       : Engineering_Units.Knots;      --WPN, PRO, IOS,
EW, NAV
end record;

-- SEND-ON-CHANGE

type Stall_Onset_Flags is (No_Stall_Onset, Stall_Onset);


--***************Function:
--*
--/ 10.6.3.2   Weight and Balance
--*
```

```
type Weight_And_Balance_Eighth_Rate is
 record
   Aircraft_Gross_Weight      : Engineering_Units.Pounds;
                                --IOS
   Electrical_Loads           : Global_Message_Types.
                                Aircraft_Electrical_Bus_Load_Array;
                                --FS
   Hydraulic_Component_Flows : Flight_Dynamics_Hydraulic_Component_Flow_Array;
                                --FS
   Fuel_Tank_Data            : Global_Message_Types.Fuel_Tank_Quantity_Array;
                                --FS
   Ownship_CG_Position        : Engineering_Units.Linear_Position_Components;
                                --FS, IOS
end record;


--***************Function:
--*
--/ 10.6.3.3   Forces and Moments
--*

type Forces_And_Moments_Eighth_Rate is
 record
   Landing_Gear_Compression_Rate : Landing_Gear_Compression_Rate_Array;
                                   --PHC
   Normal_Load_Factor            : Engineering_Units.Gravity;
                                   --FC, PHC
   Ownship_Drift_Rate        : Engineering_Units.Linear_Velocity_Components;
                                   --IOS
   Weight_On_Wheels              : Weight_On_Wheels_Array;
                                   --FC, PRO, WPN, NAV, PHC, FS
   Wheel_Status                  : Wheel_Status_Array;
                                   --FC, PHC
end record;


-- SEND-ON-CHANGE

type Touchdown_Data is
 record
```

```
   Location : Touchdown_Location;
   State    : Touchdown_State;
end record;

--***************Function:
--*
--/ 10.6.3.4   Envelope Violation
--*

--SEND-ON-CHANGE ONLY

type Structural_Crash_Flags is (No_Crash, Crash);


--***************Function:
--*
--/ 10.6.3.5   Flight Dynamics Support
--*

-- See Control_Types for responses to IOS


--*****************************************
--*
--/ 10.6.4   Flight Dynamics Representation Specs
--*
--*****************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes          : constant := 8; -- Bits per byte
   Byte_Size      : constant := 1 * Bytes;
   Halfword_Size  : constant := 2 * Bytes;
   Word_Size      : constant := 4 * Bytes;

   Angular_Position_Components_Size : constant :=
      Engineering_Units.Angular_Position_Components_Size;
   Angular_Acceleration_Components_Size : constant :=
      Engineering_Units.Angular_Acceleration_Components_Size;
```

```
Angular_Velocity_Components_Size : constant :=
   Engineering_Units.Angular_Velocity_Components_Size;
Earth_Acceleration_Components_Size : constant :=
   Engineering_Units.Earth_Acceleration_Components_Size;
Earth_Position_Components_Size : constant :=
   Engineering_Units.Earth_Position_Components_Size;
Earth_Velocity_Components_Size : constant :=
   Engineering_Units.Earth_Velocity_Components_Size;
Linear_Acceleration_Components_Size : constant :=
   Engineering_Units.Linear_Acceleration_Components_Size;
Linear_Velocity_Components_Size : constant :=
   Engineering_Units.Linear_Velocity_Components_Size;
Linear_Position_Components_Size : constant :=
   Engineering_Units.Linear_Position_Components_Size;
Vibration_Characteristics_Size : constant :=
   Global_Message_Types.Vibration_Characteristics_Size;
Aircraft_Electrical_Bus_Load_Array_Size : constant :=
   Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size;
Fuel_Tank_Quantity_Array_Size : constant :=
   Global_Message_Types.Fuel_Tank_Quantity_Array_Size;

-- 10.6.3.1
for Buffet_Status use
 record
   Buffet_Vibration at 0
                    range 0..Vibration_Characteristics_Size-1;
   Buffet_State     at Vibration_Characteristics_Size/Bytes
                    range 0..Byte_Size-1;
end record;
Buffet_Status_Size : constant :=
   Vibration_Characteristics_Size + Byte_Size;
for Buffet_Status'size use Buffet_Status_Size;

for Equations_Of_Motion_Max_Rate use
 record
   Flight_Parameters_Wind_Axis
      at 0
      range 0..Angular_Position_Components_Size-1;
   Ownship_Angular_Acceleration
      at Angular_Position_Components_Size/Bytes
      range 0..Angular_Acceleration_Components_Size-1;
```

```
Ownship_Angular_Position
   at Angular_Position_Components_Size/Bytes +
      Angular_Acceleration_Components_Size/Bytes
   range 0..Angular_Position_Components_Size-1;
Ownship_Angular_Velocity
   at Angular_Position_Components_Size/Bytes +
      Angular_Acceleration_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes
   range 0..Angular_Velocity_Components_Size-1;
Ownship_Attitude_Relative_To_Deck
   at Angular_Position_Components_Size/Bytes +
      Angular_Acceleration_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes
   range 0..Angular_Position_Components_Size-1;
Ownship_Earth_Axis_Acceleration
   at Angular_Position_Components_Size/Bytes +
      Angular_Acceleration_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes
   range 0..Earth_Acceleration_Components_Size-1;
Ownship_Earth_Axis_Position
   at Angular_Position_Components_Size/Bytes +
      Angular_Acceleration_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Earth_Acceleration_Components_Size/Bytes
   range 0..Earth_Position_Components_Size-1;
Ownship_Earth_Axis_Velocity
   at Angular_Position_Components_Size/Bytes +
      Angular_Acceleration_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Earth_Acceleration_Components_Size/Bytes +
      Earth_Position_Components_Size/Bytes
   range 0..Earth_Velocity_Components_Size-1;
Ownship_Linear_Acceleration
   at Angular_Position_Components_Size/Bytes +
```

```
         Angular_Acceleration_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Angular_Velocity_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Earth_Acceleration_Components_Size/Bytes +
         Earth_Position_Components_Size/Bytes +
         Earth_Velocity_Components_Size/Bytes
      range 0..Linear_Acceleration_Components_Size-1;
   Ownship_Linear_Velocity
      at Angular_Position_Components_Size/Bytes +
         Angular_Acceleration_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Angular_Velocity_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Earth_Acceleration_Components_Size/Bytes +
         Earth_Position_Components_Size/Bytes +
         Earth_Velocity_Components_Size/Bytes +
         Linear_Acceleration_Components_Size/Bytes
      range 0..Linear_Velocity_Components_Size-1;
   Buffet_Output
      at Angular_Position_Components_Size/Bytes +
         Angular_Acceleration_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Angular_Velocity_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Earth_Acceleration_Components_Size/Bytes +
         Earth_Position_Components_Size/Bytes +
         Earth_Velocity_Components_Size/Bytes +
         Linear_Acceleration_Components_Size/Bytes +
         Linear_Velocity_Components_Size/Bytes
      range 0..Buffet_Status_Size-1;
end record;
for Equations_Of_Motion_Max_Rate'size use
   Angular_Position_Components_Size +
   Angular_Acceleration_Components_Size +
   Angular_Position_Components_Size +
   Angular_Velocity_Components_Size +
   Angular_Position_Components_Size +
   Earth_Acceleration_Components_Size +
   Earth_Position_Components_Size +
   Earth_Velocity_Components_Size +
```

```
      Linear_Acceleration_Components_Size +
      Linear_Velocity_Components_Size +
      Buffet_Status_Size;

   for Equations_Of_Motion_Quarter_Rate use
    record
      Equivalent_Airspeed at  0 range 0..Word_Size-1;
      Mach_Number         at  4 range 0..Word_Size-1;
      Ground_Speed        at  8 range 0..Word_Size-1;
      True_Airspeed       at 12 range 0..Word_Size-1;
   end record;
   for Equations_Of_Motion_Quarter_Rate'size use
      4 * Word_Size;

   for Stall_Onset_Flags'size use 8;

   -- 10.6.3.2
   Number_Of_Flight_Dynamics_Hydraulic_Components : constant :=
      Global_Message_Types.Flight_Dynamics_Hydraulic_Component'pos (
      Global_Message_Types.Flight_Dynamics_Hydraulic_Component'last) -
      Global_Message_Types.Flight_Dynamics_Hydraulic_Component'pos (
      Global_Message_Types.Flight_Dynamics_Hydraulic_Component'first) +
1;

   Flight_Dynamics_Hydraulic_Component_Flow_Array_Size : constant :=
      Number_Of_Flight_Dynamics_Hydraulic_Components * Word_Size;
   for Flight_Dynamics_Hydraulic_Component_Flow_Array'size use
      Flight_Dynamics_Hydraulic_Component_Flow_Array_Size;

   for Weight_And_Balance_Eighth_Rate use
    record
      Aircraft_Gross_Weight
         at 0
         range 0..Word_Size-1;
      Electrical_Loads
         at Word_Size/Bytes
         range 0..Aircraft_Electrical_Bus_Load_Array_Size-1;
      Hydraulic_Component_Flows
         at Word_Size/Bytes +
            Aircraft_Electrical_Bus_Load_Array_Size/Bytes
        range 0..Flight_Dynamics_Hydraulic_Component_Flow_Array_Size-1;
      Fuel_Tank_Data
```

```
      at Word_Size/Bytes +
         Aircraft_Electrical_Bus_Load_Array_Size/Bytes +
         Flight_Dynamics_Hydraulic_Component_Flow_Array_Size/Bytes
      range 0..Fuel_Tank_Quantity_Array_Size-1;
   Ownship_CG_Position
      at Word_Size/Bytes +
         Aircraft_Electrical_Bus_Load_Array_Size/Bytes +
         Flight_Dynamics_Hydraulic_Component_Flow_Array_Size/Bytes +
         Fuel_Tank_Quantity_Array_Size/Bytes
      range 0..Linear_Position_Components_Size-1;
end record;
for Weight_And_Balance_Eighth_Rate'size use
   Word_Size +
   Aircraft_Electrical_Bus_Load_Array_Size +
   Flight_Dynamics_Hydraulic_Component_Flow_Array_Size +
   Fuel_Tank_Quantity_Array_Size +
   Linear_Position_Components_Size;

-- 10.6.3.3
Number_Of_Landing_Gear : constant :=
   Global_Message_Types.Aircraft_Landing_Gear'pos (
   Global_Message_Types.Aircraft_Landing_Gear'last) -
   Global_Message_Types.Aircraft_Landing_Gear'pos (
   Global_Message_Types.Aircraft_Landing_Gear'first) + 1;

Landing_Gear_Compression_Rate_Array_Size : constant :=
   Number_Of_Landing_Gear * Word_Size;
for Landing_Gear_Compression_Rate_Array'size use
   Landing_Gear_Compression_Rate_Array_Size;

Weight_On_Wheels_Array_Size : constant :=
   Number_Of_Landing_Gear * Byte_Size;
for Weight_On_Wheels_Array'size use
   Weight_On_Wheels_Array_Size;

for Wheel_Status_Data use
 record
   Wheel_Speed     at 0                     range 0..Word_Size-1;
   Tire_Slip_Angle at 1 * Word_Size/Bytes range 0..Word_Size-1;
   Tire_Skid       at 2 * Word_Size/Bytes range 0..Byte_Size-1;
   -- 3 bytes spare
```

```
end record;
Wheel_Status_Data_Size : constant := 3 * Word_Size;
for Wheel_Status_Data'size use Wheel_Status_Data_Size;

Wheel_Status_Array_Size : constant :=
   Number_Of_Landing_Gear * Wheel_Status_Data_Size;
for Wheel_Status_Array'size use Wheel_Status_Array_Size;

for Forces_And_Moments_Eighth_Rate use
 record
   Landing_Gear_Compression_Rate
      at  0
      range 0..Landing_Gear_Compression_Rate_Array_Size-1;
   Normal_Load_Factor
      at Landing_Gear_Compression_Rate_Array_Size/Bytes
      range 0..Word_Size-1;
   Ownship_Drift_Rate
      at Landing_Gear_Compression_Rate_Array_Size/Bytes +
         Word_Size/Bytes
      range 0..Linear_Velocity_Components_Size-1;
   Weight_On_Wheels
      at Landing_Gear_Compression_Rate_Array_Size/Bytes +
         Word_Size/Bytes +
         Linear_Velocity_Components_Size/Bytes
      range 0..Weight_On_Wheels_Array_Size-1;
   -- one byte spare
   Wheel_Status
      at Landing_Gear_Compression_Rate_Array_Size/Bytes +
         Word_Size/Bytes +
         Linear_Velocity_Components_Size/Bytes +
         Weight_On_Wheels_Array_Size/Bytes +
         Byte_Size/Bytes
      range 0..Wheel_Status_Array_Size-1;
end record;
for Forces_And_Moments_Eighth_Rate'size use
   Landing_Gear_Compression_Rate_Array_Size +
   Word_Size +
   Linear_Velocity_Components_Size +
   Weight_On_Wheels_Array_Size +
   Byte_Size +
   Wheel_Status_Array_Size;
```

```
    for Touchdown_Location'size use 8;

    for Touchdown_Data use
     record
       Location at 0 range 0..Byte_Size-1;
       State    at 1 range 0..Byte_Size-1;
    end record;
    for Touchdown_Data'size use 2 * Bytes;

    -- 10.6.3.4
    for Structural_Crash_Flags'size use 8;

end Flight_Dynamics_Output_Interface_Types;
```

```
----------------------------------------------------------------------------
-- %Z% Unit Name:        Flight_Dynamics_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
----------------------------------------------------------------------------
--
-- Purpose:
--   This package specifies all the message objects which are sent by the
--   Flight_Dynamics segment.
--
-- Adaptation:
--   The first step in adaptation is to determine which of the functions
--   in this segment will not be performed, based on simulator
requirements.
--   The messages associated with these functions need not be sent, and
--   should therefore be deleted or commented out.
--
--   Each message declaration is followed by a comment line containing
--   "Destination:" and the abbreviations of the segment(s) whic  receive
--   this message.  These comments should be modified to acco...  or
--   (a) the presence or absence of other segments, and (b) the
requirements
--  of the other segments for data.  For example, if segment X is absent,
--   then the notation that segment X is a destination of a given
--   message should be removed.  Similarly, when segment Y does not
require
--   the data in a given message, then the notation that segment Y is a
--   destination for that message should be removed.
--
--   When the segment abbreviations have all been removed for a message,
--   it is clear that this message need not be sent, and the message
--   object declaration itself may be commented out or deleted.
--
with Flight_Dynamics_Output_Interface_Types;
with Control_Types;
```

```
--
package Flight_Dynamics_Output_Interface is
--
--**************************************************************
--*                                                            *
--/ 10.7   Flight Dynamics Output Interface                    *
--*                                                            *
--**************************************************************


--***************Function:
--*
--/ 10.7.1   Equations of Motion
--*

Equations_Of_Motion_Max_Rate_Outputs :
   Flight_Dynamics_Output_Interface_Types.
   Equations_Of_Motion_Max_Rate;
--
-- Destination: ENV, EW,FC,IOS,NAV,PHC,PRO,RDR,VIS,WPN
--


Equations_Of_Motion_Quarter_Rate_Outputs :
   Flight_Dynamics_Output_Interface_Types.
   Equations_Of_Motion_Quarter_Rate;
--
-- Destination: ENV, EW, FC, IOS, NAV, PHC, PRO, WPN
--

-- SEND-ON-CHANGE OUTPUTS

Stall_Onset :
   Flight_Dynamics_Output_Interface_Types.
   Stall_Onset_Flags;
--
-- Destination: NAV
--


--***************Function:
--*
```

```
--/ 10.7.2   Weight and Balance
--*

Weight_And_Balance_Eighth_Rate_Outputs :
   Flight_Dynamics_Output_Interface_Types.
   Weight_And_Balance_Eighth_Rate;
--
-- Destination: FS, IOS, NAV
--


--***************Function:
--*
--/ 10.7.3   Forces and Moments
--*

Forces_And_Moments_Eighth_Rate_Outputs :
   Flight_Dynamics_Output_Interface_Types.
   Forces_And_Moments_Eighth_Rate;
--
-- Destination: FC, FS, NAV, PHC, PRO, WPN
--

-- SEND-ON-CHANGE OUTPUTS

Touchdown_Message :
   Flight_Dynamics_Output_Interface_Types.
   Touchdown_Data;
--
-- Destination: VIS, IOS, FS, FC
--


--***************Function:
--*
--/ 10.7.4   Envelope Violation
--*

--SEND-ON-CHANGE OUTPUTS

Ownship_Structural_Crash :
```

```
   Flight_Dynamics_Output_Interface_Types.
   Structural_Crash_Flags;
--
-- Destination: ENV, IOS
--


--***************Function:
--*
--/ 10.7.5   Flight Dynamics Support
--*

-- SEND-ON-CHANGE OUTPUTS

Flight_Dynamics_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Flight_Dynamics_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Flight_Dynamics_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Flight_Dynamics_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--
```

```
Flight_Dynamics_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Flight_Dynamics_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Flight_Dynamics_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--


--*
end Flight_Dynamics_Output_Interface;
```

```
-------------------------------------------------------------------------
-- %Z% Unit Name:        Propulsion_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Propulsion segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Propulsion_Output_Interface_Types is
--
--*************************************************************
--*                                                           *
--/ 10.8   Propulsion Output Interface Types                  *
--*                                                           *
--*************************************************************

--****************************************
```

```
--*
--/ 10.8.1   Aircraft/Simulator Specific Propulsion Types
--*
--*****************************************

-- Declare all D/Os in the flight station that this segment will
-- turn on or off
type Propulsion_Discrete_Outputs is (
   EEC_Caution_Light,
   BUC_Caution_Light,
   JFS_Run_Light,
   Hyd_Oil_Pressure_Warning_Light,
   Engine_Warning_Light,
   Engine_Fire_Warning_Light,
   Overheat_Caution_Light,
   EPU_Generator_Fail_Light,
   EPU_Pmg_Fail_Light,
   Main_Generator_Fail_Light,
   EPU_Run_Light,
   EPU_Hydrazn_Light,
   EPU_Air_Light,
   EPU_Fired);


--*****************************************
--*
--/ 10.8.2   Aircraft/Simulator Reusable Propulsion Types
--*
--*****************************************

type Engine_Inlet_Area_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Engineering_Units.Normalized;
     --0.0 = fully closed
     --1.0 = fully opened

type Temperature_Pressure_Data is
 record
   Temperature : Engineering_Units.Degrees_C;
   Pressure    : Engineering_Units.PSI;
end record;
```

```
type Engine_Temperature_Pressure_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Temperature_Pressure_Data;

type Compressor_Data is
 record
   Inlet_Fan_Speed               : Engineering_Units.RPM;
   Low_Speed_Compressor_Speed    : Engineering_Units.RPM;
   High_Speed_Compressor_Speed : Engineering_Units.RPM;
end record;

type Engine_Compressor_Data_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Compressor_Data;

type Engine_Vibration_Characteristics_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Global_Message_Types.Vibration_Characteristics;

type Engine_Turbine_Data is
 record
   Turbine_Fan_Speed           : Engineering_Units.RPM;
   Turbine_Inlet_Temperature : Engineering_Units.Degrees_C;
end record;

type Engine_Turbine_Data_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Engine_Turbine_Data;

type Engine_Thrust_Data is
 record
   Thrust                      : Engineering_Units.Pounds;
   Thrust_Reverser_Position : Engineering_Units.Normalized;
    --0.0 = fully retracted
    --1.0 = fully extended
end record;

type Engine_Thrust_Data_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Engine_Thrust_Data;
```

```
type Fired_State is (Fired, Not_Fired);

type Fired_Array is array (Global_Message_Types.Aircraft_Engine)
   of Fired_State;

type Engine_Gear_Data is
 record
   Accessory_Gearbox_Speed : Engineering_Units.RPM;
   Constant_Drive_Speed    : Engineering_Units.RPM;
   PTO_Shaft_Speed         : Engineering_Units.RPM;
end record;

type Engine_Gear_Data_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Engine_Gear_Data;

type APU_Data is
 record
   Actual_Fuel_Flow        : Engineering_Units.Lbs_Per_Hour;
   Shaft_Speed             : Engineering_Units.RPM;
   Exhaust_Gas_Temperature : Engineering_Units.Degrees_C;
   Oil                    : Global_Message_Types.Fluid_Characteristics;
   Bleed_Air               : Temperature_Pressure_Data;
end record;

type APU_Data_Array is array (Global_Message_Types.Aircraft_APU)
   of APU_Data;

type Fuel_Flow_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Engineering_Units.Lbs_Per_Hour;

type Engine_Nozzle_Data is
 record
   Nozzle_Area             : Engineering_Units.Normalized;
    --0.0 =  fully closed
    --1.0 =  fully opened
   Exhaust_Gas_Temperature : Engineering_Units.Degrees_C;
   EPR                     : Base_Types.Float_32;
end record;
```

```
type Engine_Nozzle_Data_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Engine_Nozzle_Data;

type Engine_Fluid_Characteristics_Array is
   array (Global_Message_Types.Aircraft_Engine)
   of Global_Message_Types.Fluid_Characteristics;

type Propulsion_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types.Propulsion_Hydraulic_Component)
   of Engineering_Units.Gal_Per_Min;

Number_Of_Propulsion_Discrete_Outputs : constant :=
   Propulsion_Discrete_Outputs'pos (Propulsion_Discrete_Outputs'last) -
   Propulsion_Discrete_Outputs'pos (Propulsion_Discrete_Outputs'first)
+ 1;


--******************************************
--*
--/ 10.8.3   Propulsion Segment Output Records
--*
--******************************************


--***************Function:
--*
--/ 10.8.3.1   Engine Inlet
--*

type Engine_Inlet_System_Quarter_Rate is
 record
   Engine_Inlet_Area                : Engine_Inlet_Area_Array;
                                      --FS,IOS
  Engine_Inlet_Temperature_Pressure : Engine_Temperature_Pressure_Array;
                                      --FS,IOS
end record;


--***************Function:
--*
```

```
--/ 10.8.3.2    Core Engine
--*

type Core_Engine_Half_Rate is
 record
   Engine_Compressor : Engine_Compressor_Data_Array; --FS,IOS,PHC
   Engine_Probes     : Engine_Temperature_Pressure_Array; --FS
   Engine_Vibration  : Engine_Vibration_Characteristics_Array;--PHC
   Engine_Turbine    : Engine_Turbine_Data_Array; --FS,IOS,PHC
end record;


--***************Function:
--*
--/ 10.8.3.3    Thrust Generation
--*

type Thrust_Generation_Quarter_Rate is
 record
   Engine_Thrust : Engine_Thrust_Data_Array;--FD, IOS, PHC
end record;

-- See 10.8.3.2 above for definition of Fired_Array


--***************Function:
--*
--/ 10.8.3.4    Starting System
--*

--SEND-ON-CHANGE ONLY

-- See 10.8.3.2 above for definition of Fired_Array


--***************Function:
--*
--/ 10.8.3.5    Engine Bleed Air System
--*

type Bleed_Air_System_Quarter_Rate is
```

```
 record
   Engine_Bleed_Air   : Engine_Temperature_Pressure_Array; --FS, IOS
   Engine_Anti_Icing : Engine_Temperature_Pressure_Array; --FS, IOS
end record;


--***************Function:
--*
--/ 10.8.3.6   Transmission System
--*

type Transmission_System_Quarter_Rate is
 record
   Engine_Gear : Engine_Gear_Data_Array;           --FS
end record;


--***************Function:
--*
--/ 10.8.3.7   Auxiliary Power Unit System
--*

type Auxiliary_Power_Unit_Quarter_Rate is
 record
   APU : APU_Data_Array;        --FS, PHC, IOS
end record;


--***************Function:
--*
--/ 10.8.3.8   Engine Fuel System
--*

type Engine_Fuel_System_Quarter_Rate is
 record
   Actual_Engine_Fuel_Flow : Fuel_Flow_Array; --FS, IOS
end record;


--***************Function:
--*
```

```
--/ 10.8.3.9   Engine Exhaust System
--*

type Engine_Nozzle_Half_Rate is
 record
   Engine_Nozzle : Engine_Nozzle_Data_Array;       --FS
end record;


--***************Function:
--*
--/ 10.8.3.10   Engine Oil System
--*

type Engine_Oil_System_Eighth_Rate is
 record
   Engine_Oil : Engine_Fluid_Characteristics_Array; --FS,IOS
end record;


--***************Function:
--*
--/ 10.8.3.11   Propulsion Support
--*

type Propulsion_Support_Sixteenth_Rate is
 record
   Electrical_Loads           : Global_Message_Types.
                                Aircraft_Electrical_Bus_Load_Array;--FS
  Hydraulic_Component_Flows : Propulsion_Hydraulic_Component_Flow_Array;--FS
end record;

-- SEND-ON-CHANGE

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Propulsion_Discrete_Output_And_State is
record
   Name  : Propulsion_Discrete_Outputs;
```

```
   State : Base_Types.Discrete_State;
end record;

--
-- ...or it will collect the discretes into an array;
--
subtype Propulsion_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32
   range 1..Number_Of_Propulsion_Discrete_Outputs;

type Propulsion_Discrete_Output_Array is array (
   Propulsion_Discrete_Output_Count) of
   Propulsion_Discrete_Output_And_State;


--
-- ...and send the ones which have changed in one of these messages:
--
type Propulsion_Discrete_Output_List is
record
   Number_Of_DOs    : Propulsion_Discrete_Output_Count;
   Discrete_Outputs : Propulsion_Discrete_Output_Array;
end record;

-- See Control_Types for responses to IOS

--******************************************
--*
--/ 10.8.4   Propulsion Representation Specs
--*
--******************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes     : constant := 8; -- Bits per byte
   Byte_Size : constant := 1 * Bytes;
   Word_Size : constant := 4 * Bytes;

   Number_Of_Engines : constant :=
      Global_Message_Types.Aircraft_Engine'pos (
```

```
   Global_Message_Types.Aircraft_Engine'last) -
   Global_Message_Types.Aircraft_Engine'pos (
   Global_Message_Types.Aircraft_Engine'first) + 1;

-- 10.8.3.1
Engine_Inlet_Area_Array_Size : constant :=
   Number_Of_Engines * Word_Size;
for Engine_Inlet_Area_Array'size use
   Engine_Inlet_Area_Array_Size;

for Temperature_Pressure_Data use
 record
   Temperature at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Pressure    at 1 * Word_Size/Bytes range 0..Word_Size-1;
end record;
Temperature_Pressure_Data_Size : constant :=
   2 * Word_Size;
for Temperature_Pressure_Data'size use
   Temperature_Pressure_Data_Size;

Engine_Temperature_Pressure_Array_Size : constant :=
   Temperature_Pressure_Data_Size * Number_Of_Engines;
for Engine_Temperature_Pressure_Array'size use
   Engine_Temperature_Pressure_Array_Size;

for Engine_Inlet_System_Quarter_Rate use
 record
   Engine_Inlet_Area
      at 0
      range 0..Engine_Inlet_Area_Array_Size-1;
   Engine_Inlet_Temperature_Pressure
      at Engine_Inlet_Area_Array_Size/Bytes
      range 0..Engine_Temperature_Pressure_Array_Size-1;
end record;
for Engine_Inlet_System_Quarter_Rate'size use
   Engine_Inlet_Area_Array_Size +
   Engine_Temperature_Pressure_Array_Size;

-- 10.8.3.2
for Compressor_Data use
 record
```

```
   Inlet_Fan_Speed
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Low_Speed_Compressor_Speed
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
   High_Speed_Compressor_Speed
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
end record;
Compressor_Data_Size : constant := 3 * Word_Size;
for Compressor_Data'size use Compressor_Data_Size;

Engine_Compressor_Data_Array_Size : constant :=
   Compressor_Data_Size * Number_Of_Engines;
for Engine_Compressor_Data_Array'size use
   Engine_Compressor_Data_Array_Size;

Vibration_Characteristics_Size : constant :=
   Global_Message_Types.Vibration_Characteristics_Size;

Engine_Vibration_Characteristics_Array_Size : constant :=
   Vibration_Characteristics_Size * Number_Of_Engines;
for Engine_Vibration_Characteristics_Array'size use
   Engine_Vibration_Characteristics_Array_Size;

for Engine_Turbine_Data use
 record
   Turbine_Fan_Speed
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Turbine_Inlet_Temperature
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
end record;
Engine_Turbine_Data_Size : constant := 2 * Word_Size;
for Engine_Turbine_Data'size use Engine_Turbine_Data_Size;

Engine_Turbine_Data_Array_Size : constant :=
   Engine_Turbine_Data_Size * Number_Of_Engines;
for Engine_Turbine_Data_Array'size use
   Engine_Turbine_Data_Array_Size;

for Core_Engine_Half_Rate use
 record
   Engine_Compressor
```

```
        at 0
        range 0..Engine_Compressor_Data_Array_Size-1;
      Engine_Probes
        at Engine_Compressor_Data_Array_Size/Bytes
        range 0..Engine_Temperature_Pressure_Array_Size-1;
      Engine_Vibration
        at Engine_Compressor_Data_Array_Size/Bytes +
           Engine_Temperature_Pressure_Array_Size/Bytes
        range 0..Engine_Vibration_Characteristics_Array_Size-1;
      Engine_Turbine
        at Engine_Compressor_Data_Array_Size/Bytes +
           Engine_Temperature_Pressure_Array_Size/Bytes +
           Engine_Vibration_Characteristics_Array_Size/Bytes
        range 0..Engine_Turbine_Data_Array_Size-1;
   end record;
   for Core_Engine_Half_Rate'size use
      Engine_Compressor_Data_Array_Size +
      Engine_Temperature_Pressure_Array_Size +
      Engine_Vibration_Characteristics_Array_Size +
      Engine_Turbine_Data_Array_Size;

   -- 10.8.3.3
   for Engine_Thrust_Data use
    record
      Thrust
        at 0 * Word_Size/Bytes range 0..Word_Size-1;
      Thrust_Reverser_Position
        at 1 * Word_Size/Bytes range 0..Word_Size-1;
   end record;
   Engine_Thrust_Data_Size : constant := 2 * Word_Size;
   for Engine_Thrust_Data'size use Engine_Thrust_Data_Size;

   Engine_Thrust_Data_Array_Size : constant :=
      Engine_Thrust_Data_Size * Number_Of_Engines;
   for Engine_Thrust_Data_Array'size use
      Engine_Thrust_Data_Array_Size;

   for Thrust_Generation_Quarter_Rate'size use
      Engine_Thrust_Data_Array_Size;

   Fired_State_Size : constant := 8;
```

```
for Fired_State'size use Fired_State_Size;

for Fired_Array'size use
   Fired_State_Size * Number_Of_Engines;

-- 10.8.3.5
for Bleed_Air_System_Quarter_Rate use
 record
   Engine_Bleed_Air
      at 0
      range 0..Engine_Temperature_Pressure_Array_Size-1;
   Engine_Anti_Icing
      at Engine_Temperature_Pressure_Array_Size/Bytes
      range 0..Engine_Temperature_Pressure_Array_Size-1;
end record;
for Bleed_Air_System_Quarter_Rate'size use
   2 * Engine_Temperature_Pressure_Array_Size;

-- 10.8.3.6
for Engine_Gear_Data use
 record
   Accessory_Gearbox_Speed
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Constant_Drive_Speed
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
   PTO_Shaft_Speed
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
end record;
Engine_Gear_Data_Size : constant :=
   3 * Word_Size;
for Engine_Gear_Data'size use
   Engine_Gear_Data_Size;

Engine_Gear_Data_Array_Size : constant :=
   Engine_Gear_Data_Size * Number_Of_Engines;
for Engine_Gear_Data_Array'size use
   Engine_Gear_Data_Array_Size;

for Transmission_System_Quarter_Rate'size use
   Engine_Gear_Data_Array_Size;
```

```
-- 10.8.3.7
Number_Of_APUs : constant :=
   Global_Message_Types.Aircraft_APU'pos (
   Global_Message_Types.Aircraft_APU'last) -
   Global_Message_Types.Aircraft_APU'pos (
   Global_Message_Types.Aircraft_APU'first) + 1;

Fluid_Characteristics_Size : constant :=
   Global_Message_Types.Fluid_Characteristics_Size;

for APU_Data use
 record
   Actual_Fuel_Flow
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Shaft_Speed
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
   Exhaust_Gas_Temperature
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
   Oil
      at 3 * Word_Size/Bytes range 0..Fluid_Characteristics_Size-1;
   Bleed_Air
      at 3 * Word_Size/Bytes + Fluid_Characteristics_Size/Bytes
      range 0..Temperature_Pressure_Data_Size-1;
end record;

APU_Data_Size : constant :=
   3 * Word_Size +
   Temperature_Pressure_Data_Size +
   Fluid_Characteristics_Size;
for APU_Data'size use APU_Data_Size;

APU_Data_Array_Size : constant :=
   APU_Data_Size * Number_Of_APUs;
for APU_Data_Array'size use APU_Data_Array_Size;

for Auxiliary_Power_Unit_Quarter_Rate'size use APU_Data_Array_Size;

-- 10.8.3.8
Fuel_Flow_Array_Size : constant :=
   Word_Size * Number_Of_Engines;
for Fuel_Flow_Array'size use Fuel_Flow_Array_Size;
```

```
for Engine_Fuel_System_Quarter_Rate'size use Fuel_Flow_Array_Size;

-- 10.8.3.9
for Engine_Nozzle_Data use
 record
   Nozzle_Area
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Exhaust_Gas_Temperature
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
   EPR
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
end record;

Engine_Nozzle_Data_Size : constant :=
   3 * Word_Size;
for Engine_Nozzle_Data'size use Engine_Nozzle_Data_Size;

Engine_Nozzle_Data_Array_Size : constant :=
   Engine_Nozzle_Data_Size * Number_Of_Engines;
for Engine_Nozzle_Data_Array'size use
   Engine_Nozzle_Data_Array_Size;

for Engine_Nozzle_Half_Rate'size use
   Engine_Nozzle_Data_Array_Size;

-- 10.8.3.10
Engine_Fluid_Characteristics_Array_Size : constant :=
   Fluid_Characteristics_Size * Number_Of_Engines;

for Engine_Fluid_Characteristics_Array'size use
   Engine_Fluid_Characteristics_Array_Size;

for Engine_Oil_System_Eighth_Rate'size use
   Engine_Fluid_Characteristics_Array_Size;

-- 10.8.3.11
Number_Of_Propulsion_Hydraulic_Components : constant :=
   Global_Message_Types.Propulsion_Hydraulic_Component'pos (
   Global_Message_Types.Propulsion_Hydraulic_Component'last) -
   Global_Message_Types.Propulsion_Hydraulic_Component'pos (
```

```
   Global_Message_Types.Propulsion_Hydraulic_Component'first) + 1;

Propulsion_Hydraulic_Component_Flow_Array_Size : constant :=
   Word_Size * Number_Of_Propulsion_Hydraulic_Components;

Aircraft_Electrical_Bus_Load_Array_Size : constant :=
   Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size;

for Propulsion_Support_Sixteenth_Rate use
 record
   Electrical_Loads
      at 0
      range 0..Aircraft_Electrical_Bus_Load_Array_Size-1;
   Hydraulic_Component_Flows
      at Aircraft_Electrical_Bus_Load_Array_Size/Bytes
      range 0..Propulsion_Hydraulic_Component_Flow_Array_Size-1;
end record;
for Propulsion_Support_Sixteenth_Rate'size use
   Aircraft_Electrical_Bus_Load_Array_Size +
   Propulsion_Hydraulic_Component_Flow_Array_Size;

Propulsion_Discrete_Output_And_State_Size : constant := 2 * Bytes;
for Propulsion_Discrete_Output_And_State'size use
   Propulsion_Discrete_Output_And_State_Size;

Propulsion_Discrete_Output_Array_Size : constant :=
   Propulsion_Discrete_Output_And_State_Size *
   Number_Of_Propulsion_Discrete_Outputs;
for Propulsion_Discrete_Output_Array'size use
   Propulsion_Discrete_Output_Array_Size;

for Propulsion_Discrete_Output_List use
 record
   Number_Of_DOs     at 0
                     range 0..Word_Size-1;
   Discrete_Outputs  at Word_Size/Bytes
                     range 0..
                     Propulsion_Discrete_Output_Array_Size-1;
 end record;
for Propulsion_Discrete_Output_List'size use
   Word_Size + Propulsion_Discrete_Output_Array_Size;
```

```
--*
end Propulsion_Output_Interface_Types;
```

```
---------------------------------------------------------------------------
-- %Z% Unit Name:       Propulsion_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
---------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Propulsion segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
-- of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
with Propulsion_Output_Interface_Types;
with Control_Types;
```

```
--
package Propulsion_Output_Interface is
--
--**************************************************************
--*                                                            *
--/ 10.9   Propulsion Output Interface                         *
--*                                                            *
--**************************************************************


--***************Function:
--*
--/ 10.9.1   Engine Inlet System
--*

Engine_Inlet_System_Quarter_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Engine_Inlet_System_Quarter_Rate;
--
-- Destination:  FS, IOS
--


--***************Function:
--*
--/ 10.9.2   Core Engine
--*

Core_Engine_Half_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Core_Engine_Half_Rate;
--
-- Destination: FS, IOS, PHC
--


--***************Function:
--*
--/ 10.9.3   Thrust Generation
--*
```

```
Thrust_Generation_Quarter_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Thrust_Generation_Quarter_Rate;
--
-- Destination:  ENV, FD, FS, IOS, PHC
--

--SEND-ON-CHANGE OUTPUTS

Afterburner_Fired :
   Propulsion_Output_Interface_Types.
   Fired_Array;
--
-- Destination:  ENV, FD, FS, PHC
--


--***************Function:
--*
--/ 10.9.4   Starting System
--*

--SEND-ON-CHANGE OUTPUTS

Engine_Fired :
   Propulsion_Output_Interface_Types.
   Fired_Array;
--
-- Destination:  FD, FS, PHC
--


--***************Function:
--*
--/ 10.9.5   Engine Bleed Air System
--*

Engine_Bleed_Air_System_Quarter_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Bleed_Air_System_Quarter_Rate;
--
```

```
-- Destination:  FS, IOS
--


--***************Function:
--*
--/ 10.9.6   Transmission System
--*

Transmission_System_Quarter_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Transmission_System_Quarter_Rate;
--
-- Destination:  FD, FS
--


--***************Function:
--*
--/ 10.9.7   Auxiliary Power Unit System
--*

Auxiliary_Power_Unit_System_Quarter_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Auxiliary_Power_Unit_Quarter_Rate;
--
-- Destination:  FS, IOS,PHC
--


--***************Function:
--*
--/ 10.9.8   Engine Fuel System
--*

Engine_Fuel_System_Quarter_Rate_Output :
   Propulsion_Output_Interface_Types.
   Engine_Fuel_System_Quarter_Rate;
--
-- Destination:  FS, IOS
--
```

```
--***************Function:
--*
--/ 10.9.9   Engine Exhaust System
--*

Engine_Exhaust_System_Half_Rate_Output :
   Propulsion_Output_Interface_Types.
   Engine_Nozzle_Half_Rate;
--
-- Destination :  FS
--


--***************Function:
--*
--/ 10.9.10   Engine Oil System
--*

Engine_Oil_System_Eighth_Rate_Output :
   Propulsion_Output_Interface_Types.
   Engine_Oil_System_Eighth_Rate;
--
-- Destination:  FS, IOS
--


--***************Function:
--*
--/ 10.9.11   Propulsion Support
--*

Propulsion_Support_Sixteenth_Rate_Outputs :
   Propulsion_Output_Interface_Types.
   Propulsion_Support_Sixteenth_Rate;
--
-- Destination:  FS
--

-- SEND-ON-CHANGE OUTPUTS
```

```
Propulsion_Discrete_Output_Change :
   Propulsion_Output_Interface_Types.
   Propulsion_Discrete_Output_List;
--
-- Destination:  FS
--

Propulsion_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Propulsion_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Propulsion_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Propulsion_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Propulsion_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
```

```
--

Propulsion_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Propulsion_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--
--*
end Propulsion_Output_Interface;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Navigation_Communication_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Navigation_Communication segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Navigation_Communication_Output_Interface_Types is
--
--*************************************************************
--*                                                           *
--/ 10.10   Navigation/Communication Output Interface Types   *
--*                                                           *
--*************************************************************

--****************************************
```

```
--*
--/ 10.10.1   Aircraft/Simulator Specific
--/               Navigation/Communication Types
--*
--*******************************************

-- Number of UHF, VHF and Guard radios in the aircraft
Max_UHF_Radios   : constant := 1;
Max_VHF_Radios   : constant := 1;
Max_Guard_Radios : constant := 1;

-- Maximum length of a radio station call sign in this simulation
Max_ID_Length    : constant := 6;

-- Maximum destinations in the simulation database
Max_Destinations : constant := 10;


-- How many (and which) crew stations have TACANs
subtype TACAN_Crew_Stations is Global_Message_Types.Crew_Station range
   Global_Message_Types.Pilot..Global_Message_Types.Pilot;

-- How many (and which) crew stations have AHRSs
subtype AHRS_Crew_Stations is Global_Message_Types.Crew_Station range
   Global_Message_Types.Pilot..Global_Message_Types.Pilot;

-- Declare all D/Os in the flight station that this segment will
-- turn on or off
type Navigation_Communication_Discrete_Outputs is (
   Compass_Warning_Flag,           --FS
   Power_Adequacy_Indicator,       --FS
   Accelerometer_Valid,            --FS
   Attitude_Warning_Flag,          --FS
   Coarse_Align_Light_Enable,      --FS
   Fine_Align_Light_Enable,        --FS
   Gyro_Valid,                     --FS, FC
   INS_Attitude_Valid,             --FS, FC, WPN,RDR,EW
   NAV_Ready_Light_Enable,         --FS
   Platform_Heading_Warning,       --FS, FC
   INS_Navigation_Valid,           --FS, FC, WPN,RDR,EW
   Navigation_Warning_Horn_Enable, --FS
```

```
Low_Altitude_Warning,               --FS
RDR_Alt_Sef_Test_Fail,              --FS
Glideslope_Capture,                 --FS
Glideslope_Valid,                   --IOS, FS, FC
ILS_Audio_Enable,                   --FS
Inner_Beacon_Light_Enable,          --FS
Localizer_Capture_Flag,             --FS
Localizer_Valid,                    --FS
Marker_Beacon_Audio_Enable,         --FS
Middle_Beacon_Light_Enable,         --FS
Outer_Beacon_Light_Enable,          --FS
TACAN_A_A_Rec,                      --FS
TACAN_A_A_T_R,                      --FS
TACAN_Receive,                      --FS
TACAN_T_R,                          --FS
Power_Off_Warning,                  --FS
TACAN_To_From_Indicator,            --FS
TACAN_Transmitting,                 --EW, RDR
TACAN_Self_Test_Fail,               --FS
TACAN_Audio_Enable,
TACAN_Bearing_Valid,
TACAN_Range_Valid,
TACAN_To_From_Out_Of_View,
TACAN_Range_Shutter,
UHF_Guard_Valid,
UHF_Receive_Enable,
Guard_Receive_Enable,
UHF_Transmit_Enable,
VHF_Receive_Enable,
VHF_Transmit_Enable,
Mission_Launch_Tone_Enable,
Threat_Warning_Tone_Enable,
IFF_Caution_Light,
IFF_Test_Light_Enable,
IFF_Transmitting,
ADC_Light_Enable,
CADC_Valid,
Le_Flaps_Light_Enable,
Master_Caution_Light_Enable,
Stby_Gains_Light_Enable,
Takeoff_Landing_Config_Warning,
```

```
   Radio_Transmitting_UHF,          --RDR, EW
   Radio_Transmitting_VHF,          --RDR, EW
   IFF_Mode_4_Tone_Enable,                 --FS
   Mode_4_Reply_Light_Enable,       --FS
   Angle_Of_Attack_Lights_Red,      --FS
   Angle_Of_Attack_Lights_Amber,      --FS
   Angle_Of_Attack_Lights_Green,      --FS
   Stall_Warning_Enable,             --FS, FC
   Alt_Pneu_Hide_Flag,
   ADI_Off_Hide_Flag,
   ADI_Aux_Hide_Flag,
   ADI_Glideslope_Hide_Flag,
   ADI_Localizer_Hide_Flag,
   HSI_Deviation_Hide_Flag,
   HSI_DME_Center_Shutter_Hide,
   HSI_Off_Hide_Flag,
   HSI_To_Arrow_Flag,
   HSI_From_Arrow_Flag,
   Marker_Beacon_Light,
   Standby_ADI_Off_Hide_Flag,
   Avionics_Caution_Warning_Light,
   Radar_Alt_Caution_Warning_Light,
   Equip_Hot_Caution_Warning_Light,
   ADC_Caution_Warning_Light,
   CADC_Caution_Warning_Light);

--******************************************
--*
--/ 10.10.2  Aircraft/Simulator Reusable
--/              Navigation/Communication Types
--*
--******************************************


--*************************
--*
--/ 10.10.2.1   AHRS Types
--*
--*************************

type AHRS_Indicator_Data is
```

```
 record
   AHRS_Indicated_Attitude : Engineering_Units.Angular_Position_Components;--FS
   AHRS_Magnetic_Heading    : Engineering_Units.Degrees; --FS,EW,RDR
   AHRS_Gyro_Heading        : Engineering_Units.Degrees; --FS,EW,RDR
end record;

type AHRS_Indicator_Data_Array is array (AHRS_Crew_Stations)
   of AHRS_Indicator_Data;

--**************************
--*
--/ 10.10.2.2   INS Types
--*
--**************************

subtype Destination_Type is Base_Types.Unsigned_Integer_16 range
   1..Max_Destinations;

subtype Waypoint_Display_Data_Count is
   Base_Types.Unsigned_Integer_16 range 1..
   Global_Message_Types.Maximum_Number_Of_Waypoints;

type Waypoint_Display_Data is
   array (Waypoint_Display_Data_Count) of Base_Types.Sim_Boolean;

type INS_Position_Update_Type is (HUD, Radar);

--**************************
--*
--/ 10.10.2.3   ILS And Radio Types
--*
--**************************

type Beacon_Type is (
   Outer,
   Middle,
   Inner,
   Back_Course,
   Fan,
   Bone,
   Zone);
```

```
subtype Call_Sign is String (1..Max_ID_Length);


--**************************
--*
--/ 10.10.2.4   TACAN Types
--*
--**************************

type TACAN_Position_Data is
 record
   Relative_Bearing : Engineering_Units.Degrees;      -- FC, FS, EW, IOS
   Deviation        : Engineering_Units.Degrees;      -- FS, FC, IOS
   DME_Range_Rate   : Engineering_Units.Feet_Per_Sec; -- FS, IOS
   DME_Slant_Range  : Engineering_Units.Feet;         -- FS, IOS
end record;

type TACAN_Call_Sign_Type is (TACAN, VORTAC);

type TACAN_Indicator_Data is
 record
   Selected_Channel  : Engineering_Units.Hertz;   -- FS, IOS
   Station_Call_Sign : Call_Sign;                 -- FS
   Call_Sign_Type    : TACAN_Call_Sign_Type;      -- FS
   Jamming_Level     : Engineering_Units.Decibel; -- FS
end record;

type TACAN_Position_Data_Array is
   array (TACAN_Crew_Stations) of TACAN_Position_Data;

type TACAN_Indicator_Data_Array is
   array (TACAN_Crew_Stations) of TACAN_Indicator_Data;

--**************************
--*
--/ 10.10.2.5   UHF/VHF/HF/Intercom Types
--*
--**************************

type Radio_Indicator_Data is
```

```
 record
   Radio_Selected_Frequency          : Engineering_Units.Hertz;    --
FS, IOS
   Tuned_Station_Morse_Code_Call_Sign : Call_Sign;                 -- FS
   Marker_Beacon_Code                : Call_Sign;                  -- FS
   Noise_Level                       : Engineering_Units.Decibel;  -- FS
   Jamming_Level                     : Engineering_Units.Decibel;  -- FS
end record;

subtype UHF_Radio_Indicator_Array_Count is
 Base_Types.Unsigned_Integer_16 range 1..Max_UHF_Radios;
subtype VHF_Radio_Indicator_Array_Count is
 Base_Types.Unsigned_Integer_16 range 1..Max_VHF_Radios;
subtype Guard_Radio_Indicator_Array_Count is
 Base_Types.Unsigned_Integer_16 range 1..Max_Guard_Radios;

type UHF_Radio_Indicator_Data_Array is
   array (UHF_Radio_Indicator_Array_Count) of Radio_Indicator_Data;

type VHF_Radio_Indicator_Data_Array is
   array (VHF_Radio_Indicator_Array_Count) of Radio_Indicator_Data;

type Guard_Radio_Indicator_Data_Array is
   array (Guard_Radio_Indicator_Array_Count) of Radio_Indicator_Data;


--*************************
--*
--/ 10.10.2.6   Discrete Types
--*
--*************************

Number_Of_Navigation_Communication_Discrete_Outputs : constant :=
 Navigation_Communication_Discrete_Outputs'pos (
    Navigation_Communication_Discrete_Outputs'last) -
 Navigation_Communication_Discrete_Outputs'pos (
    Navigation_Communication_Discrete_Outputs'first) + 1;


--****************************************
--*
--/ 10.10.3   Navigation/Communication Segment Output Records
```

```
--*
--*******************************************


--***************Function:
--*
--/ 10.10.3.1   Attitude Heading Reference System (AHRS)
--*

type AHRS_Quarter_Rate is
record
   AHRS_Indicators : AHRS_Indicator_Data_Array; -- FS,EW,RDR
end record;

--***************Function:
--*
--/ 10.10.3.2   Inertial Navigation System (INS)
--*

type INS_Half_Rate is
 record
   INS_Aircraft_Position  : Engineering_Units.Earth_Position_Components;
                              -- WPN, EW, FC, RDR, FS, IOS
   INS_Indicated_Attitude : Engineering_Units.Angular_Position_Components;
                              -- FC, RDR, FS, EW, IOS
    Apparent_Wander_Angle  : Engineering_Units.Degrees;
                              -- RDR
    Indicated_Slip         : Engineering_Units.Radians;
                              -- FC, FS, RDR
    INS_Drift_Angle        : Engineering_Units.Degrees;
                              -- FC, FS, RDR
    INS_Ground_Speed       : Engineering_Units.Knots;
                              -- FC, FS, RDR
   Sensed_Attitude_Rates  : Engineering_Units.Angular_Velocity_Components;
                              -- FS,FC
   Sensed_Velocity       : Engineering_Units.Linear_Velocity_Components;
                               -- FS, RDR
   Standby_Attitude      : Engineering_Units.Angular_Position_Components;
                              -- FC, FS, EW
    INS_Pitch_Steering     : Engineering_Units.Radians;
                              -- FS, FC
```

```
    INS_Roll_Steering        : Engineering_Units.Radians;
                               -- FS, FC
    INS_Wind_Speed           : Engineering_Units.Knots;
                               -- FS
    INS_Wind_Direction       : Engineering_Units.Degrees;
                               -- FS
   Sensed_Acceleration    : Engineering_Units.Linear_Acceleration_Components;
                               -- FS
    INS_Time_Tag             : Base_Types.Unsigned_Integer_32;
                               -- FS
    INS_Status               : Base_Types.Unsigned_Integer_32;
                              -- FS
end record;

type INS_Quarter_Rate is
 record
   INS_Course_Deviation     : Engineering_Units.Degrees;  -- IOS, FS, FC
   INS_Cross_Track_Distance : Engineering_Units.Feet;      -- FS
   INS_Desired_Track        : Engineering_Units.Degrees;   -- FS
   INS_True_Heading         : Engineering_Units.Degrees;   -- FS, EW
   INS_Magnetic_Heading     : Engineering_Units.Degrees;   -- FS, EW
end record;

type INS_Eighth_Rate is
 record
   Beacon_Bearing                    : Engineering_Units.Degrees;
-- FS
   Beacon_Range                      : Engineering_Units.Nautical_Miles;
-- FS
   Beacon_Time_Delay                 : Engineering_Units.Seconds;
-- FS
   Bearing_From_Ip_To_Target         : Engineering_Units.Degrees;
  -- FS
   Elevation_From_Beacon_To_Target : Engineering_Units.Feet;
  -- FS
   Elevation_From_Ip_To_Target       : Engineering_Units.Feet;
  -- FS
   Indicated_Rate_Of_Turn            : Engineering_Units.Radians_Per_Sec;
-- FS
   INS_Align_Time                    : Engineering_Units.Hours;
-- FS
   INS_Height_Above_Terrain          : Engineering_Units.Feet;
  -- FS
```

```
   INS_Magnetic_Variation              : Engineering_Units.Degrees;
  -- FS
   Optimum_Cruise_Altitude             : Engineering_Units.Feet;
  -- FS
   Optimum_Mach_Number                 : Engineering_Units.Mach_Range;
-- FS
   Position_Update_Delta               : Engineering_Units.Feet;
  -- FS
   Range_From_Ip_To_Target             : Engineering_Units.Nautical_Miles;
-- FS
   Range_To_Steerpoing                 : Engineering_Units.Nautical_Miles;
-- FS
   Selected_OAP_Bearing                : Engineering_Units.Degrees;
-- FS
  Selected_OAP_Position             : Engineering_Units.Lat_Long_Location;--
FS
   Selected_OAP_Range                  : Engineering_Units.Nautical_Miles;
-- FS
   Selected_TACAN_Bearing              : Engineering_Units.Degrees;
  -- FS
   Selected_TACAN_Range                : Engineering_Units.Nautical_Miles;
-- FS
   Distance_To_Next_Waypoint           : Engineering_Units.Nautical_Miles;
-- IOS,FS
   Time_To_Next_Waypoint               : Base_Types.Signed_Integer_32;
-- IOS,FS
   INU_Align_Status                    : Base_Types.Unsigned_Integer_32;
-- FS
   Destination_Number                  : Destination_Type;
  -- FS
   Next_Waypoint_Number              : Global_Message_Types.INS_Waypoints;
-- IOS,FS
   Waypoint_Display_Selected           : Waypoint_Display_Data;
  -- FS
end record;

-- SEND-ON-CHANGE

type INS_Position_Update_Data is
 record
   Position_Fix_Request : Base_Types.Sim_Boolean;    -- RDR
   Position_Update_Type : INS_Position_Update_Type; -- RDR
end record;
```

```
-- See Global_Message_Types for declaration of Waypoint_Change


--***************Function:
--*
--/ 10.10.3.3   Radar Altimeter
--*

type Radar_Alt_Eighth_Rate is
 record
   Radar_Altitude       : Engineering_Units.Feet;         -- FC, FS, EW
   Altitude_Rate        : Engineering_Units.Feet_Per_Sec; -- FC, FS, EW
   Low_Altitude_Setting : Engineering_Units.Feet;         -- FS
   Altitude_Reliable    : Base_Types.Sim_Boolean;         -- FC, FS, EW
 end record;

--***************Function:
--*
--/ 10.10.3.4   Radio Navigation Aid System
--*


--***************Subfunction:
--*
--/ 10.10.3.4.1   Instrument Landing System (ILS)
--*

type ILS_Half_Rate is
 record
   Glideslope_Deviation      : Engineering_Units.Degrees; -- IOS, FS, FC
   ILS_Selected_Frequency    : Engineering_Units.Hertz;   -- IOS, FS
   Localizer_Deviation       : Engineering_Units.Degrees; -- IOS, FS, FC
   Marker_Station_Call_Sign  : Call_Sign;                 -- FS
   ILS_Station_Call_Sign     : Call_Sign;                 -- FS
   Marker_Station_Type       : Beacon_Type;               -- FS
end record;

--***************Subfunction:
--*
--/ 10.10.3.4.2   TACAN
--*
```

```
type TACAN_Quarter_Rate is
record
   TACAN_Position : TACAN_Position_Data_Array; -- FS, FC, EW, IOS
end record;

type TACAN_Eighth_Rate is
record
   TACAN_Indicators : TACAN_Indicator_Data_Array; --FS,IOS
end record;

--***************Subfunction:
--*
--/ 10.10.3.4.3   Microwave Landing System (MLS)
--*

--NONE


--***************Subfunction:
--*
--/ 10.10.3.4.4   Automatic Direction Finder (ADF)
--*

--NONE


--***************Subfunction:
--*
--/ 10.10.3.4.5   Global Positioning System (GPS)
--*

--NONE


--***************Subfunction:
--*
--/ 10.10.3.4.6   VOR
--*

--NONE
```

```
--***************Subfunction:
--*
--/ 10.10.3.4.7   LORAN
--*

--NONE

--***************Subfunction:
--*
--/ 10.10.3.4.8   OMEGA
--*

--NONE


--***************Subfunction:
--*
--/ 10.10.3.4.9   Station Keeping Equipment (SKE)
--*

--NONE


--***************Function:
--*
--/ 10.10.3.5   Communications
--*

--***************Subfunction:
--*
--/ 10.10.3.5.1   UHF/VHF/HF/Intercom
--*

type UHF_VHF_HF_Intercom_Eighth_Rate is
 record
   UHF_Radio_Indicators   : UHF_Radio_Indicator_Data_Array;   -- FS, IOS
   VHF_Radio_Indicators   : VHF_Radio_Indicator_Data_Array;   -- FS, IOS
   Guard_Radio_Indicators : Guard_Radio_Indicator_Data_Array; -- FS, IOS
end record;
```

```
--***************Subfunction:
--*
--/ 10.10.3.5.2   SATCOM
--*

--NONE


--***************Subfunction:
--*
--/ 10.10.3.5.3   Joint Tactical Information
--/               Distribution System (JTIDS)
--*

--NONE


--***************Function:
--*
--/ 10.10.3.6   Identification Friend or Foe (IFF)
--*

type IFF_Eighth_Rate is
record
  Mode_C_Altitude              : Engineering_Units.Feet;          -- IOS
  IFF_Selection                : Global_Message_Types.IFF_Data; -- FS
  Interrogation_Response_Code : Global_Message_Types.IFF_Code;
--FS,IOS,EW,RDR
end record;

--***************Function:
--*
--/ 10.10.3.7   Star Tracker
--*


--***************Function:
--*
--/ 10.10.3.8   Doppler Radar
--*
```

```
--****************Function:
--*
--/ 10.10.3.9   Air Data System (ADS)
--*

type ADS_Half_Rate is
 record
   ADC_Ambient_Temperature         : Engineering_Units.Degrees_F;     --
PRO, FS
   CADC_Mach                       : Engineering_Units.Mach_Range;    --
PRO, IOS, FS
   CADC_True_Airspeed              : Engineering_Units.Knots;         --
FS, FC
   CADC_Altitude_Error             : Engineering_Units.Feet;          --
FC
   CADC_Altitude_Rate              : Engineering_Units.Feet_Per_Sec; --
FS, FC
   CADC_AOA_Error                  : Engineering_Units.Radians;       --
FC
   CADC_Leading_Edge_Flaps_Command : Engineering_Units.Radians;       --
FC
   CADC_Pitch_Command              : Engineering_Units.Radians;       --
FC
   CADC_Roll_Command               : Engineering_Units.Radians;       --
FC
   Gun_Compensation_Yaw_Command    : Engineering_Units.Radians;       --
FC
   ADC_Angle_Of_Attack             : Engineering_Units.Radians;       --
FS, FC
   CADC_Dynamic_Pressure           : Engineering_Units.PSI;           --
FS, FC
   CADC_Indicated_Airspeed         : Engineering_Units.Knots;         --
FS, FC, IOS
   Static_Pressure                 : Engineering_Units.PSI;           --
FS, FC
   Pressure_Ratio                  : Base_Types.Float_32;             --
FS
   Air_Density_Ratio               : Base_Types.Float_32;             --
FS
   CADC_Status                     : Base_Types.Unsigned_Integer_16; --
FS
end record;
```

```
type ADS_Eighth_Rate is
 record
   ADC_Barometric_Pressure_Altitude : Engineering_Units.Feet;      --
FS, EW
   ADC_Calibrated_Airspeed         : Engineering_Units.Knots;      -- FS
   Baro_Altimeter_Setting           : Engineering_Units.Inches_Hg; --
IOS
   Baro_Reference_Altitude         : Engineering_Units.Feet;       -- FS
end record;


--***************Function:
--*
--/ 10.10.3.10   Navigation Support
--*

type HUD_Symbology_Max_Rate is
 record
  Fpm             : Engineering_Units.Polar_Direction;
  Diamond         : Engineering_Units.Polar_Direction;
  Steering_Circle : Engineering_Units.Polar_Direction;
end record;

type Command_Steering_Max_Rate is
 record
   ADI_Horizontal_Dev   : Engineering_Units.Signed_Degrees; -- FS
   ADI_Vertical_Dev     : Engineering_Units.Signed_Degrees; -- FS
   Course_Hdg           : Engineering_Units.Degrees;        -- FS
   Bearing              : Engineering_Units.Degrees;        -- FS
   Rng                  : Engineering_Units.Nautical_Miles; -- FS
   HSI_Course_Deviation : Engineering_Units.Signed_Degrees; -- FS
end record;

type Navigation_Support_Eighth_Rate is
record
  Electrical_Loads : Global_Message_Types.Aircraft_Electrical_Bus_Load_Array;
-- FS
end record;

-- See Moving_Model_Types for definition of Emitter_Unique_Data
```

```
-- SEND-ON-CHANGE

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Navigation_Communication_Discrete_Output_And_State is
record
  Name  : Navigation_Communication_Discrete_Outputs;
  State : Base_Types.Discrete_State;
end record;

--
-- ...or it will collect the discretes into an array;
--
subtype Navigation_Communication_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32 range
   1..Number_Of_Navigation_Communication_Discrete_Outputs;

type Navigation_Communication_Discrete_Output_Array is array (
   Navigation_Communication_Discrete_Output_Count) of
   Navigation_Communication_Discrete_Output_And_State;

--
-- ...and send the ones which have changed in one of these messages:
--
type Navigation_Communication_Discrete_Output_List is
record
   Number_Of_DOs     : Navigation_Communication_Discrete_Output_Count;
   Discrete_Outputs  : Navigation_Communication_Discrete_Output_Array;
end record;

-- See Control_Types for responses to IOS


--*****************************************
--*
--/ 10.10.4   Navigation/Communication Representation Specs
--*
--*****************************************
private
```

```
--
-- Declarations to make representation specs more readable
--
Bytes          : constant := 8; -- Bits per byte
Byte_Size      : constant := 1 * Bytes;
Halfword_Size : constant := 2 * Bytes;
Word_Size      : constant := 4 * Bytes;

Earth_Position_Components_Size : constant :=
   Engineering_Units.Earth_Position_Components_Size;
Angular_Position_Components_Size : constant :=
   Engineering_Units.Angular_Position_Components_Size;
Linear_Velocity_Components_Size : constant :=
   Engineering_Units.Linear_Velocity_Components_Size;
Angular_Velocity_Components_Size : constant :=
   Engineering_Units.Angular_Velocity_Components_Size;
Linear_Acceleration_Components_Size : constant :=
   Engineering_Units.Linear_Velocity_Components_Size;

-- 10.10.3.1
for AHRS_Indicator_Data use
 record
   AHRS_Indicated_Attitude
      at 0
      range 0..Angular_Position_Components_Size-1;
   AHRS_Magnetic_Heading
      at Angular_Position_Components_Size/Bytes
      range 0..Word_Size-1;
   AHRS_Gyro_Heading
      at Angular_Position_Components_Size/Bytes +
      Word_Size/Bytes
      range 0..Word_Size-1;
end record;
AHRS_Indicator_Data_Size : constant :=
   Angular_Position_Components_Size + Word_Size + Word_Size;

for AHRS_Indicator_Data'size use AHRS_Indicator_Data_Size;

Number_Of_AHRS_Crew_Stations : constant :=
   AHRS_Crew_Stations'pos (AHRS_Crew_Stations'last) -
```

```
   AHRS_Crew_Stations'pos (AHRS_Crew_Stations'first) + 1;

AHRS_Indicator_Data_Array_Size : constant :=
   Number_Of_AHRS_Crew_Stations * AHRS_Indicator_Data_Size;

for AHRS_Indicator_Data_Array'size use
   AHRS_Indicator_Data_Array_Size;

for AHRS_Quarter_Rate'size use AHRS_Indicator_Data_Array_Size;

-- 10.10.3.2
for INS_Half_Rate use
 record
   INS_Aircraft_Position
      at 0
      range 0..Earth_Position_Components_Size-1;
   INS_Indicated_Attitude
      at Earth_Position_Components_Size/Bytes
      range 0..Angular_Position_Components_Size-1;
   Apparent_Wander_Angle
      at Earth_Position_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes
      range 0..Word_Size-1;
   Indicated_Slip
      at Earth_Position_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes
      range 0..Word_Size-1;
   INS_Drift_Angle
      at Earth_Position_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes
      range 0..Word_Size-1;
   INS_Ground_Speed
      at Earth_Position_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes
      range 0..Word_Size-1;
```

```
Sensed_Attitude_Rates
   at Earth_Position_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes
   range 0..Angular_Velocity_Components_Size-1;
Sensed_Velocity
   at Earth_Position_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes
   range 0..Linear_Velocity_Components_Size-1;
Standby_Attitude
   at Earth_Position_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes +
      Linear_Velocity_Components_Size/Bytes
   range 0..Angular_Position_Components_Size-1;
INS_Pitch_Steering
   at Earth_Position_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Angular_Velocity_Components_Size/Bytes +
      Linear_Velocity_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes
   range 0..Word_Size-1;
INS_Roll_Steering
   at Earth_Position_Components_Size/Bytes +
      Angular_Position_Components_Size/Bytes +
```

```
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Angular_Velocity_Components_Size/Bytes +
            Linear_Velocity_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
      INS_Wind_Speed
         at Earth_Position_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Angular_Velocity_Components_Size/Bytes +
            Linear_Velocity_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
      INS_Wind_Direction
         at Earth_Position_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Angular_Velocity_Components_Size/Bytes +
            Linear_Velocity_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
      Sensed_Acceleration
         at Earth_Position_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes +
            Word_Size/Bytes +
            Word_Size/Bytes +
```

```
         Word_Size/Bytes +
         Word_Size/Bytes +
         Angular_Velocity_Components_Size/Bytes +
         Linear_Velocity_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes
      range 0..Linear_Acceleration_Components_Size-1;
   INS_Time_Tag
      at Earth_Position_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Angular_Velocity_Components_Size/Bytes +
         Linear_Velocity_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Linear_Acceleration_Components_Size/Bytes
      range 0..Word_Size-1;
   INS_Status
      at Earth_Position_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Angular_Velocity_Components_Size/Bytes +
         Linear_Velocity_Components_Size/Bytes +
         Angular_Position_Components_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes +
         Linear_Acceleration_Components_Size/Bytes +
```

```
            Word_Size/Bytes
         range 0..Word_Size-1;
   end record;
   for INS_Half_Rate'size use
      Earth_Position_Components_Size +
      Angular_Position_Components_Size +
      Word_Size +
      Word_Size +
      Word_Size +
      Word_Size +
      Angular_Velocity_Components_Size +
      Linear_Velocity_Components_Size +
      Angular_Position_Components_Size +
      Word_Size +
      Word_Size +
      Word_Size +
      Word_Size +
      Linear_Acceleration_Components_Size +
      Word_Size +
      Word_Size;

   for INS_Quarter_Rate use
    record
      INS_Course_Deviation     at 0 * Word_Size/Bytes range
0..Word_Size-1;
      INS_Cross_Track_Distance at 1 * Word_Size/Bytes range
0..Word_Size-1;
      INS_Desired_Track        at 2 * Word_Size/Bytes range
0..Word_Size-1;
      INS_True_Heading         at 3 * Word_Size/Bytes range
0..Word_Size-1;
      INS_Magnetic_Heading     at 4 * Word_Size/Bytes range
0..Word_Size-1;
   end record;
   for INS_Quarter_Rate'size use 5 * Word_Size;

   Waypoint_Display_Data_Size : constant :=
     Global_Message_Types.Maximum_Number_Of_Waypoints * Byte_Size;
   for Waypoint_Display_Data'size use
      Waypoint_Display_Data_Size;

   Lat_Long_Location_Size : constant :=
```

```
      Engineering_Units.Lat_Long_Location_Size;

   INS_Waypoints_Size : constant :=
      Global_Message_Types.INS_Waypoints_Size;

   for INS_Eighth_Rate use
    record
      Beacon_Bearing                   at  0 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Beacon_Range                     at  1 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Beacon_Time_Delay                at  2 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Bearing_From_Ip_To_Target        at  3 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Elevation_From_Beacon_To_Target  at  4 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Elevation_From_Ip_To_Target      at  5 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Indicated_Rate_Of_Turn           at  6 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      INS_Align_Time                   at  7 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      INS_Height_Above_Terrain         at  8 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      INS_Magnetic_Variation           at  9 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Optimum_Cruise_Altitude          at 10 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Optimum_Mach_Number              at 11 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Position_Update_Delta            at 12 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Range_From_Ip_To_Target          at 13 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Range_To_Steerpoing              at 14 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Selected_OAP_Bearing             at 15 * Word_Size/Bytes
                                       range 0..Word_Size-1;
      Selected_OAP_Position            at 16 * Word_Size/Bytes
                                     range 0..Lat_Long_Location_Size-1;
```

```
Selected_OAP_Range              at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
                                range 0..Word_Size-1;
Selected_TACAN_Bearing          at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   1 * Word_Size/Bytes
                                range 0..Word_Size-1;
Selected_TACAN_Range            at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   2 * Word_Size/Bytes
                                range 0..Word_Size-1;
Distance_To_Next_Waypoint       at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   3 * Word_Size/Bytes
                                range 0..Word_Size-1;
Time_To_Next_Waypoint           at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   4 * Word_Size/Bytes
                                range 0..Word_Size-1;
INU_Align_Status                at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   5 * Word_Size/Bytes
                                range 0..Word_Size-1;
Destination_Number              at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   6 * Word_Size/Bytes
                                range 0..Halfword_Size-1;
Next_Waypoint_Number            at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
                                   6 * Word_Size/Bytes +
                                     Halfword_Size/Bytes
                                range 0..INS_Waypoints_Size-1;
Waypoint_Display_Selected       at 16 * Word_Size/Bytes +
                                   Lat_Long_Location_Size/Bytes
+
```

```
                                    6 * Word_Size/Bytes +
                                        Halfword_Size/Bytes +
                                        INS_Waypoints_Size/Bytes
                              range 0..Waypoint_Display_Data_Size-1;
end record;
for INS_Eighth_Rate'size use
   16 * Word_Size +
   Lat_Long_Location_Size +
   6 * Word_Size +
   Halfword_Size +
   INS_Waypoints_Size +
   Waypoint_Display_Data_Size;

INS_Position_Update_Type_Size : constant := 8;
for INS_Position_Update_Type'size use
   INS_Position_Update_Type_Size;

for INS_Position_Update_Data use
 record
   Position_Fix_Request
      at 0
      range 0..Byte_Size-1;
   Position_Update_Type
      at Byte_Size/Bytes
      range 0..INS_Position_Update_Type_Size-1;
end record;
for INS_Position_Update_Data'size use
   Byte_Size + INS_Position_Update_Type_Size;

-- 10.10.3.3
for Radar_Alt_Eighth_Rate use
 record
   Radar_Altitude       at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Altitude_Rate        at 1 * Word_Size/Bytes range 0..Word_Size-1;
   Low_Altitude_Setting at 2 * Word_Size/Bytes range 0..Word_Size-1;
   Altitude_Reliable    at 3 * Word_Size/Bytes range 0..Byte_Size-1;
 end record;
for Radar_Alt_Eighth_Rate'size use 3 * Word_Size + Byte_Size;

-- 10.10.3.4
Call_Sign_Size : constant := Max_ID_Length * Bytes;
```

```
Beacon_Type_Size : constant := Byte_Size;
for Beacon_Type'size use Beacon_Type_Size;

for ILS_Half_Rate use
 record
   Glideslope_Deviation
      at 0
      range 0..Word_Size-1;
   ILS_Selected_Frequency
      at Word_Size/Bytes
      range 0..Word_Size-1;
   Localizer_Deviation
      at 2 * Word_Size/Bytes
      range 0..Word_Size-1;
   Marker_Station_Call_Sign
      at 3 * Word_Size/Bytes
      range 0..Call_Sign_Size-1;
   ILS_Station_Call_Sign
      at 3 * Word_Size/Bytes + Call_Sign_Size/Bytes
      range 0..Call_Sign_Size-1;
   Marker_Station_Type
      at 3 * Word_Size/Bytes + 2 * Call_Sign_Size/Bytes
      range 0..Beacon_Type_Size-1;
end record;
for ILS_Half_Rate'size use
   3 * Word_Size + 2 * Call_Sign_Size + Beacon_Type_Size;

Number_Of_TACAN_Crew_Stations : constant :=
   TACAN_Crew_Stations'pos (TACAN_Crew_Stations'last) -
   TACAN_Crew_Stations'pos (TACAN_Crew_Stations'first) + 1;

for TACAN_Position_Data use
 record
   Relative_Bearing at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Deviation        at 1 * Word_Size/Bytes range 0..Word_Size-1;
   DME_Range_Rate   at 2 * Word_Size/Bytes range 0..Word_Size-1;
   DME_Slant_Range  at 3 * Word_Size/Bytes range 0..Word_Size-1;
end record;
TACAN_Position_Data_Size : constant := 4 * Word_Size;
for TACAN_Position_Data'size use TACAN_Position_Data_Size;
```

```
TACAN_Position_Data_Array_Size : constant :=
   Number_Of_TACAN_Crew_Stations * TACAN_Position_Data_Size;
for TACAN_Position_Data_Array'size use
   TACAN_Position_Data_Array_Size;

for TACAN_Quarter_Rate'size use
   TACAN_Position_Data_Array_Size;

TACAN_Call_Sign_Type_Size : constant := Byte_Size;
for TACAN_Call_Sign_Type'size use TACAN_Call_Sign_Type_Size;

for TACAN_Indicator_Data use
 record
   Selected_Channel
      at 0
      range 0..Word_Size-1;
   Jamming_Level
      at Word_Size/Bytes
      range 0..Word_Size-1;
   Station_Call_Sign
      at 2 * Word_Size/Bytes
      range 0..Call_Sign_Size-1;
   Call_Sign_Type
      at 2 * Word_Size/Bytes + Call_Sign_Size/Bytes
      range 0..TACAN_Call_Sign_Type_Size-1;
end record;
TACAN_Indicator_Data_Size : constant :=
   2 * Word_Size + Call_Sign_Size + TACAN_Call_Sign_Type_Size;
for TACAN_Indicator_Data'size use TACAN_Indicator_Data_Size;

TACAN_Indicator_Data_Array_Size : constant :=
   TACAN_Indicator_Data_Size * Number_Of_TACAN_Crew_Stations;
for TACAN_Indicator_Data_Array'size use
   TACAN_Indicator_Data_Array_Size;

for TACAN_Eighth_Rate'size use TACAN_Indicator_Data_Array_Size;

-- 10.10.3.5
for Radio_Indicator_Data use
 record
```

```
   Radio_Selected_Frequency
      at 0
      range 0..Word_Size-1;
   Noise_Level
      at 1 * Word_Size/Bytes
      range 0..Word_Size-1;
   Jamming_Level
      at 2 * Word_Size/Bytes
      range 0..Word_Size-1;
   Tuned_Station_Morse_Code_Call_Sign
      at 3 * Word_Size/Bytes
      range 0..Call_Sign_Size-1;
   Marker_Beacon_Code
      at 3 * Word_Size/Bytes +
         Call_Sign_Size/Bytes
      range 0..Call_Sign_Size-1;
end record;
Radio_Indicator_Data_Size : constant :=
   3 * Word_Size + 2 * Call_Sign_Size;
for Radio_Indicator_Data'size use
   Radio_Indicator_Data_Size;

UHF_Radio_Indicator_Data_Array_Size : constant :=
   Radio_Indicator_Data_Size * Max_UHF_Radios;
for UHF_Radio_Indicator_Data_Array'size use
   UHF_Radio_Indicator_Data_Array_Size;

VHF_Radio_Indicator_Data_Array_Size : constant :=
   Radio_Indicator_Data_Size * Max_VHF_Radios;
for VHF_Radio_Indicator_Data_Array'size use
   VHF_Radio_Indicator_Data_Array_Size;

Guard_Radio_Indicator_Data_Array_Size : constant :=
   Radio_Indicator_Data_Size * Max_Guard_Radios;
for Guard_Radio_Indicator_Data_Array'size use
   Guard_Radio_Indicator_Data_Array_Size;

for UHF_VHF_HF_Intercom_Eighth_Rate use
 record
   UHF_Radio_Indicators
      at 0
```

```
            range 0..UHF_Radio_Indicator_Data_Array_Size-1;
        VHF_Radio_Indicators
            at UHF_Radio_Indicator_Data_Array_Size/Bytes
            range 0..VHF_Radio_Indicator_Data_Array_Size-1;
        Guard_Radio_Indicators
            at UHF_Radio_Indicator_Data_Array_Size/Bytes +
               VHF_Radio_Indicator_Data_Array_Size/Bytes
            range 0..Guard_Radio_Indicator_Data_Array_Size-1;
    end record;
    for UHF_VHF_HF_Intercom_Eighth_Rate'size use
        UHF_Radio_Indicator_Data_Array_Size +
        VHF_Radio_Indicator_Data_Array_Size +
        Guard_Radio_Indicator_Data_Array_Size;

    -- 10.10.3.6

    IFF_Data_Size : constant :=
        Global_Message_Types.IFF_Data_Size;

    IFF_Code_Size : constant :=
        Global_Message_Types.Max_IFF_Code_Length * Bytes;

    for IFF_Eighth_Rate use
    record
      Mode_C_Altitude
         at 0 range 0..Word_Size-1;
      IFF_Selection
         at Word_Size/Bytes range 0..IFF_Data_Size-1;
      Interrogation_Response_Code
         at Word_Size/Bytes + IFF_Data_Size/Bytes
         range 0..IFF_Code_Size-1;
    end record;
    for IFF_Eighth_Rate'size use
       Word_Size + IFF_Data_Size + IFF_Code_Size;

    -- 10.10.3.9
    for ADS_Half_Rate use
     record
       ADC_Ambient_Temperature
          at  0 * Word_Size/Bytes range 0..Word_Size-1;
       CADC_Mach
```

```
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
   Baro_Reference_Altitude
      at 3 * Word_Size/Bytes range 0..Word_Size-1;
end record;
for ADS_Eighth_Rate'size use 4 * Word_Size;

-- 10.10.3.10
Polar_Direction_Size : constant :=
   Engineering_Units.Polar_Direction_Size;

for HUD_Symbology_Max_Rate use
 record
  Fpm
    at 0 * Polar_Direction_Size/Bytes range 0..Polar_Direction_Size-1;
  Diamond
    at 1 * Polar_Direction_Size/Bytes range 0..Polar_Direction_Size-1;
  Steering_Circle
    at 2 * Polar_Direction_Size/Bytes range 0..Polar_Direction_Size-1;
end record;
for HUD_Symbology_Max_Rate'size use 3 * Polar_Direction_Size;

for Command_Steering_Max_Rate use
 record
   ADI_Horizontal_Dev
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   ADI_Vertical_Dev
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
   Course_Hdg
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
   Bearing
      at 3 * Word_Size/Bytes range 0..Word_Size-1;
   Rng
      at 4 * Word_Size/Bytes range 0..Word_Size-1;
   HSI_Course_Deviation
      at 5 * Word_Size/P .es range 0..Word_Size-1;
end record;
for Command_Steering_Max_Rate'size use 6 * Word_Size;

Aircraft_Electrical_Bus_Load_Array_Size : constant :=
   Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size;
```

```
         at  1 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_True_Airspeed
         at  2 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Altitude_Error
         at  3 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Altitude_Rate
         at  4 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_AOA_Error
         at  5 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Leading_Edge_Flaps_Command
         at  6 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Pitch_Command
         at  7 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Roll_Command
         at  8 * Word_Size/Bytes range 0..Word_Size-1;
      Gun_Compensation_Yaw_Command
         at  9 * Word_Size/Bytes range 0..Word_Size-1;
      ADC_Angle_Of_Attack
         at 10 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Dynamic_Pressure
         at 11 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Indicated_Airspeed
         at 12 * Word_Size/Bytes range 0..Word_Size-1;
      Static_Pressure
         at 13 * Word_Size/Bytes range 0..Word_Size-1;
      Pressure_Ratio
         at 14 * Word_Size/Bytes range 0..Word_Size-1;
      Air_Density_Ratio
         at 15 * Word_Size/Bytes range 0..Word_Size-1;
      CADC_Status
         at 16 * Word_Size/Bytes range 0..Halfword_Size-1;
   end record;
   for ADS_Half_Rate'size use 16 * Word_Size + Halfword_Size;

   for ADS_Eighth_Rate use
    record
      ADC_Barometric_Pressure_Altitude
         at 0 * Word_Size/Bytes range 0..Word_Size-1;
      ADC_Calibrated_Airspeed
         at 1 * Word_Size/Bytes range 0..Word_Size-1;
      Baro_Altimeter_Setting
```

```
   for Navigation_Support_Eighth_Rate'size use
      Aircraft_Electrical_Bus_Load_Array_Size;

   Navigation_Communication_Discrete_Output_And_State_Size : constant
:=
      2 * Bytes;
   for Navigation_Communication_Discrete_Output_And_State'size use
      Navigation_Communication_Discrete_Output_And_State_Size;

   Navigation_Communication_Discrete_Output_Array_Size : constant :=
      Navigation_Communication_Discrete_Output_And_State_Size *
      Number_Of_Navigation_Communication_Discrete_Outputs;
   for Navigation_Communication_Discrete_Output_Array'size use
      Navigation_Communication_Discrete_Output_Array_Size;

   for Navigation_Communication_Discrete_Output_List use
    record
      Number_Of_DOs    at 0
                       range 0..Word_Size-1;
      Discrete_Outputs at Word_Size/Bytes
                       range 0..
Navigation_Communication_Discrete_Output_Array_Size-1;
    end record;
   for Navigation_Communication_Discrete_Output_List'size use
      Word_Size + Navigation_Communication_Discrete_Output_Array_Size;

--*
end Navigation_Communication_Output_Interface_Types;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Navigation_Communication_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   12 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--   This package specifies all the message objects which are sent by the
--   Navigation_Communication segment.
--
-- Adaptation:
--   The first step in adaptation is to determine which of the functions
--   in this segment will not be performed, based on simulator
requirements.
--   The messages associated with these functions need not be sent, and
--   should therefore be deleted or commented out.
--
--   Each message declaration is followed by a comment line containing
--   "Destination:" and the abbreviations of the segment(s) which receive
--   this message.  These comments should be modified to account for
--   (a) the presence or absence of other segments, and (b) the
requirements
--   of the other segments for data.  For example, if segment X is absent,
--   then the notation that segment X is a destination of a given
--   message should be removed.  Similarly, when segment Y does not
require
--   the data in a given message, then the notation that segment Y is a
--   destination for that message should be removed.
--
--   When the segment abbreviations have all been removed for a message,
--   it is clear that this message need not be sent, and the message
--   object declaration itself may be commented out or deleted.
--
with Navigation_Communication_Output_Interface_Types;
with Global_Message_Types;
```

```
with Control_Types;
with Moving_Model_Types;
--
package Navigation_Communication_Output_Interface is
--
--***********************************************************
--*                                                         *
--/ 10.11   Navigation/Communication Output Interface       *
--*                                                         *
--***********************************************************


--***************Function:
--*
--/ 10.11.1   Attitude Heading Reference System (AHRS)
--*

AHRS_Quarter_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   AHRS_Quarter_Rate;
--
-- Destination: FS, EW, RDR
--


--***************Function:
--*
--/ 10.11.2   Inertial Navigation System (INS)
--*

INS_Half_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   INS_Half_Rate;
--
-- Destination: FS, EW, RDR, WPN, FC, IOS
--

INS_Quarter_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   INS_Quarter_Rate;
--
```

```
-- Destination: FS, IOS, FC, EW
--

INS_Eighth_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   INS_Eighth_Rate;
--
-- Destination: FS, IOS
--

-- SEND-ON-CHANGE OUTPUTS

Position_Update :
   Navigation_Communication_Output_Interface_Types.
   INS_Position_Update_Data;
--
-- Destination: RDR
--

Waypoint_Change_Output :
   Global_Message_Types.
   Waypoint_Change;
--
-- Destination: IOS, FS
--


--***************Function:
--*
--/ 10.11.3   Radar Altimeter
--*

Radar_Alt_Eighth_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   Radar_Alt_Eighth_Rate;
--
-- Destination: FS, FC, EW
--


--***************Function:
```

```
--*
--/ 10.11.4   Radio Navigation Aid System
--*

--***************Subfunction:
--*
--/ 10.11.4.1   Instrument Landing System (ILS)
--*

ILS_Half_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   ILS_Half_Rate;
--
-- Destination: IOS, FS, FC
--


--***************Subfunction:
--*
--/ 10.11.4.2   TACAN
--*


TACAN_Quarter_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   TACAN_Quarter_Rate;
--
-- Destination: FS, FC, EW, IOS
--

TACAN_Eighth_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   TACAN_Eighth_Rate;
--
-- Destination: FS, IOS
--


--***************Subfunction:
--*
--/ 10.11.4.3   Microwave Landing System (MLS)
```

--*

--NONE

--***************Subfunction:
--*
--/ 10.11.4.4 Automatic Direction Finder (ADF)
--*

--NONE

--***************Subfunction:
--*
--/ 10.11.4.5 Global Positioning System (GPS)
--*

--NONE

--***************Subfunction:
--*
--/ 10.11.4.6 VOR
--*

--NONE

--***************Subfunction:
--*
--/ 10.11.4.7 LORAN
--*

--NONE

--***************Subfunction:
--*
--/ 10.11.4.8 OMEGA
--*

```
--NONE


--***************Subfunction:
--*
--/ 10.11.4.9   Station Keeping Equipment (SKE)
--*

--NONE


--***************Function:
--*
--/ 10.11.5   Communications
--*


--***************Subfunction:
--*
--/ 10.11.5.1   UHF/VHF/HF/Intercom
--*

UHF_VHF_HF_Intercom_Eighth_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   UHF_VHF_HF_Intercom_Eighth_Rate;
--
-- Destination: FS, IOS
--

--***************Subfunction:
--*
--/ 10.11.5.2   SATCOM
--*

--NONE


--***************Subfunction:
--*
--/ 10.11.5.3   Joint Tactical Information
```

```
--/                    Distribution System (JTIDS)
--*

--NONE


--***************Function:
--*
--/ 10.11.6   Identification Friend or Foe (IFF)
--*


IFF_Eighth_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   IFF_Eighth_Rate;
--
-- Destination: FS, IOS, RDR, EW
--


--***************Function:
--*
--/ 10.11.7   Star Tracker
--*

--NONE


--***************Function:
--*
--/ 10.11.8   Doppler Radar
--*

--NONE


--***************Function:
--*
--/ 10.11.9   Air Data System (ADS)
--*
```

```
ADS_Half_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   ADS_Half_Rate;
--
-- Destination: FS, PRO, FC, IOS
--

ADS_Eighth_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   ADS_Eighth_Rate;
--
-- Destination: FS, IOS, EW
--


--***************Function:
--*
--/ 10.11.10  Navigation Support
--*

HUD_Symbology_Max_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   HUD_Symbology_Max_Rate;
--
-- Destination: FS
--

Command_Steering_Max_Rate_Outputs :
   Navigation_Communication_Output_Interface_Types.
   Command_Steering_Max_Rate;
--
-- Destination: FS
--

Navigation_Support_Eighth_Rate_Output :
   Navigation_Communication_Output_Interface_Types.
   Navigation_Support_Eighth_Rate;
--
-- Destination: FS
--
```

```
Navigation_Emitter_Unique_Data :
   Moving_Model_Types.
   Emitter_Unique_Data;
--
-- Destination: ENV
--

-- SEND-ON-CHANGE OUTPUTS

Navigation_Communication_Discrete_Output_Change :
   Navigation_Communication_Output_Interface_Types.
   Navigation_Communication_Discrete_Output_List;
--
-- Destination: FS, FC, WPN, RDR, EW, IOS, PRO
--

Navigation_Communication_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Navigation_Communication_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Navigation_Communication_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Navigation_Communication_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
```

```
-- Destination : IOS
--

Navigation_Communication_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Navigation_Communication_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Navigation_Communication_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--
--*
end Navigation_Communication_Output_Interface;
```

```
---------------------------------------------------------------------------
-- %Z% Unit Name:       Weapons_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  3 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
---------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Weapons segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
with Moving_Model_Types;
--
package Weapons_Output_Interface_Types is
--
--***************************************************************
--*                                                             *
--/ 10.12   Weapons Output Interface Types                      *
--*                                                             *
--***************************************************************
```

```
--****************************************
--*
--/ 10.12.1   Aircraft/Simulator Specific Weapons Types
--*
--****************************************

Maximum_Number_Of_Ownship_Weapons : constant := 10;

-- Declare all D/Os in the flight station that this segment will
-- turn on or off
type Weapons_Discrete_Outputs is
     (To_Be_Determined);


--****************************************
--*
--/ 10.12.2   Aircraft/Simulator Reusable Weapons Types
--*
--****************************************

subtype Ownship_Weapons_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Ownship_Weapons;

type Ownship_Weapons_Dynamic_Data_Array is
   array (Ownship_Weapons_Count) of
   Moving_Model_Types.Moving_Model_Dynamic_Data;

Number_Of_Weapons_Discrete_Outputs : constant :=
   Weapons_Discrete_Outputs'pos (Weapons_Discrete_Outputs'last) -
   Weapons_Discrete_Outputs'pos (Weapons_Discrete_Outputs'first) + 1;

type Weapons_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types.Weapons_Hydraulic_Component)
   of Engineering_Units.Gal_Per_Min;


--****************************************
--*
--/ 10.12.3   Weapons Segment Output Records
--*
```

```
--******************************************

--***************Function:
--*
--/ 10.12.3.1   Ownship Fire Control
--*

type Ownship_Fire_Control_Eighth_Rate is
 record
   Electrical_Loads          : Global_Message_Types.
                          Aircraft_Electrical_Bus_Load_Array;    --FS
   Hydraulic_Component_Flows : Weapons_Hydraulic_Component_Flow_Array;
--FS
end record;

-- SEND-CN-CHANGE

type Ownship_Weapon_Fire_Status is
 record
   Weapon_Fired       : Moving_Model_Types.Moving_Model_ID;  -- PHC, FD,
FS, IOS
   Intended_Target    : Moving_Model_Types.Moving_Model_ID;  -- EW, IOS
   Station_Fired_From : Global_Message_Types.Stores_Station; -- PHC, FD,
FS, IOS
end record;


--***************Function:
--*
--/ 10.12.3.2   Ownship Weapon Dynamics
--*

type Ownship_Weapon_Dynamics_Half_Rate is
 record
   Number_Of_Weapons              : Ownship_Weapons_Count;
   Ownship_Weapons_Dynamic_Data : Ownship_Weapons_Dynamic_Data_Array;
end record;     --IOS,RDR,VIS,FS,EW


--***************Function:
```

```
--*
--/ 10.12.3.3   Ownship Weapons Stores
--*

-- SEND-ON-CHANGE

-- The WPN segment may send out a message of this type which includes
-- weight and weapon loaded in a given station.  This is sent whenever
-- a station's load changes.
type Ownship_Stores_Data is
 record
   Station           : Global_Message_Types.Stores_Station;       --IOS,
FS, FD
   Stores_Type       : Global_Message_Types.Station_Weapon_Load;  --IOS,
FS, FD
   Stores_Load_Weight : Engineering_Units.Pounds;                  --FD
end record;

-- The WPN segment may send out a status for a given stores station...
type Weapon_Status_Record is
 record
   Station : Global_Message_Types.Stores_Station;    -- FS
   Status  : Global_Message_Types.Weapon_Status;     -- FS
end record;

-- ...or it may collect those individual statuses into an array...
type Weapon_Status_Array is
   array (Global_Message_Types.Stores_Station_Count)
   of Weapon_Status_Record;

-- ...and send a message containing statuses for all weapon stores
-- stations that have changed.
type Weapon_Status_Output is
 record
   Number_Of_Stations : Global_Message_Types.Stores_Station_Count;
   Data               : Weapon_Status_Array;
end record;


--***************Function:
--*
--/ 10.12.3.4   Target Designation
```

```
--*

-- NONE

--***************Function:
--*
--/ 10.12.3.5   Threat Weapons' Damage Assessment
--*

--SEND-ON-CHANGE

-- see Moving_Model_Types for definitions of Scoring_Damage_Data and
-- Scoring_Activation_Status


--***************Function:
--*
--/ 10.12.3.6   Weapons Support
--*

--SEND-ON-CHANGE

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Weapons_Discrete_Output_And_State is record
   ID    : Weapons_Discrete_Outputs;
   State : Base_Types.Discrete_State;
 end record;

--
-- ...or it will collect the discretes into an array;
--
subtype Weapons_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32
   range 1..Number_Of_Weapons_Discrete_Outputs;

type Weapons_Discrete_Output_Array is array (
   Weapons_Discrete_Output_Count) of
   Weapons_Discrete_Output_And_State;
```

```
--
-- ...and send the ones which have changed in one of these messages:
--
type Weapons_Discrete_Output_List is
 record
   Number_Of_DOs    : Weapons_Discrete_Output_Count;
   Discrete_Outputs : Weapons_Discrete_Output_Array;
 end record;

type HUD_Output_Record is
 record
   Pull_Up_Anticipation_Cue : Engineering_Units.Seconds;   -- FS
   CCIP_Reticle             : Engineering_Units.Polar_Direction;  -- FS
end record;

-- See Moving_Model_Types for definition of Moving_Model_Deactivation

-- See Global_Message_Types for definition of Jettison_Status and
-- Weapon_Station_Change

-- See Control_Types for responses to IOS


--*****************************************
--*
--/ 10.12.4   Weapons Representation Specs
--*
--*****************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes         : constant := 8; -- Bits per byte
   Byte_Size     : constant := 1 * Bytes;
   Halfword_Size : constant := 2 * Bytes;
   Word_Size     : constant := 4 * Bytes;

   -- 10.12.3.1
   Number_Of_Weapons_Hydraulic_Components : constant :=
```

```
   Global_Message_Types.Weapons_Hydraulic_Component'pos (
   Global_Message_Types.Weapons_Hydraulic_Component'last) -
   Global_Message_Types.Weapons_Hydraulic_Component'pos (
   Global_Message_Types.Weapons_Hydraulic_Component'first) + 1;

Weapons_Hydraulic_Component_Flow_Array_Size : constant :=
   Number_Of_Weapons_Hydraulic_Components * Word_Size;
for Weapons_Hydraulic_Component_Flow_Array'size use
   Weapons_Hydraulic_Component_Flow_Array_Size;

for Ownship_Fire_Control_Eighth_Rate use
 record
   Electrical_Loads
      at 0
      range 0..Global_Message_Types.
               Aircraft_Electrical_Bus_Load_Array_Size-1;
   Hydraulic_Component_Flows
      at Global_Message_Types.
         Aircraft_Electrical_Bus_Load_Array_Size/Bytes
      range 0..Weapons_Hydraulic_Component_Flow_Array_Size-1;
 end record;
for Ownship_Fire_Control_Eighth_Rate'size use
   Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size +
   Weapons_Hydraulic_Component_Flow_Array_Size;

Stores_Station_Size : constant :=
   Global_Message_Types.Stores_Station_Size;

Moving_Model_ID_Size : constant :=
   Moving_Model_Types.Moving_Model_ID_Size;

for Ownship_Weapon_Fire_Status use
 record
   Weapon_Fired
      at 0
      range 0..Moving_Model_ID_Size-1;
   Intended_Target
      at Moving_Model_ID_Size/Bytes
      range 0..Moving_Model_ID_Size-1;
   Station_Fired_From
      at 2 * Moving_Model_ID_Size/Bytes
```

```
      range 0..Stores_Station_Size-1;
end record;
for Ownship_Weapon_Fire_Status'size use
   2 * Moving_Model_ID_Size +
   Stores_Station_Size;

-- 10.12.3.2
Ownship_Weapons_Dynamic_Data_Array_Size : constant :=
   Moving_Model_Types.Moving_Model_Dynamic_Data_Size *
   Maximum_Number_Of_Ownship_Weapons;

for Ownship_Weapon_Dynamics_Half_Rate use
 record
   Number_Of_Weapons
      at 0
      range 0..Word_Size-1;
   Ownship_Weapons_Dynamic_Data
      at Word_Size/Bytes
      range 0..Ownship_Weapons_Dynamic_Data_Array_Size-1;
end record;
for Ownship_Weapon_Dynamics_Half_Rate'size use
   Word_Size +
   Ownship_Weapons_Dynamic_Data_Array_Size;

-- 10.12.3.3
Station_Weapon_Load_Size : constant :=
   Global_Message_Types.Station_Weapon_Load_Size;
Weapon_Status_Size : constant :=
   Global_Message_Types.Weapon_Status_Size;

for Ownship_Stores_Data use
 record
   Station
      at 0
      range 0..Stores_Station_Size-1;
   Stores_Type
      at Stores_Station_Size/Bytes
      range 0..Station_Weapon_Load_Size-1;
   Stores_Load_Weight
      at Stores_Station_Size/Bytes +
         Station_Weapon_Load_Size/Bytes +
```

```
          Halfword_Size/Bytes                    -- 2 Bytes spare
             range 0..Word_Size-1;
      end record;
      Ownship_Stores_Data_Size : constant :=
         Stores_Station_Size +
         Station_Weapon_Load_Size +
         Halfword_Size +
         Word_Size;
      for Ownship_Stores_Data'size use
         Ownship_Stores_Data_Size;

      for Weapon_Status_Record use
       record
         Station
            at 0 range 0..Stores_Station_Size-1;
         Status
            at Stores_Station_Size/Bytes range 0..Weapon_Status_Size-1;
      end record;
      Weapon_Status_Record_Size : constant :=
         Stores_Station_Size +
         Weapon_Status_Size;
      for Weapon_Status_Record'size use
         Weapon_Status_Record_Size;

      Weapon_Status_Array_Size : constant :=
         Weapon_Status_Record_Size *
         Global_Message_Types.Number_Of_Stores_Stations;
      for Weapon_Status_Array'size use
         Weapon_Status_Array_Size;

      for Weapon_Status_Output use
       record
         Number_Of_Stations
            at 0 range 0..Word_Size-1;
         Data
            at Word_Size/Bytes range 0..Weapon_Status_Array_Size-1;
      end record;
      for Weapon_Status_Output'size use
         Word_Size +
         Weapon_Status_Array_Size;
```

```
   -- 10.12.3.6
   Polar_Direction_Size : constant :=
      Engineering_Units.Polar_Direction_Size;
   for HUD_Output_Record use
    record
      Pull_Up_Anticipation_Cue
         at 0 range 0..Word_Size-1;
      CCIP_Reticle
         at Word_Size/Bytes range 0..Polar_Direction_Size-1;
   end record;
   for HUD_Output_Record'size use
      Word_Size +
      Polar_Direction_Size;

--*
end Weapons_Output_Interface_Types;
```

```
--------------------------------------------------------------------
-- %Z% Unit Name:        Weapons_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
--------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Weapons segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
-- of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
with Weapons_Output_Interface_Types;
with Control_Types;
```

```
with Global_Message_Types;
with Moving_Model_Types;
--
package Weapons_Output_Interface is
--
--*************************************************************
--*                                                           *
--/ 10.13   Weapons Output Interface                          *
--*                                                           *
--*************************************************************


--***************Function:
--*
--/ 10.13.1   Ownship Fire Control
--*

Ownship_Fire_Control_Eighth_Rate_Outputs :
   Weapons_Output_Interface_Types.
   Ownship_Fire_Control_Eighth_Rate;
--
-- Destination: FS
--

--SEND-ON-CHANGE OUTPUTS

Ownship_Weapon_Fire_Occurrence :
   Weapons_Output_Interface_Types.
   Ownship_Weapon_Fire_Status;
--
-- Destination: ENV, FS, FD, IOS, EW, PHC, VIS
--


--***************Function:
--*
--/ 10.13.2   Ownship Weapon Dynamics
--*

Ownship_Weapon_Dynamics_Half_Rate_Outputs :
   Weapons_Output_Interface_Types.
```

```
   Ownship_Weapon_Dynamics_Half_Rate;
--
-- Destination:  ENV, RDR, IOS, VIS, FS, EW, NAV
--


--***************Function:
--*
--/ 10.13.3   Ownship Weapons Stores
--*

--SEND-ON-CHANGE OUTPUTS

Ownship_Stores_Data_Update :
   Weapons_Output_Interface_Types.
   Ownship_Stores_Data;
--
-- Destination: ENV, FD, IOS, FS
--

Ownship_Stores_Status_Update :
   Weapons_Output_Interface_Types.
   Weapon_Status_Output;
--
-- Destination: ENV, FD, IOS, FS
--


--***************Function:
--*
--/ 10.13.4   Target Designation
--*

-- NONE


--***************Function:
--*
--/ 10.13.5   Threat Weapons' Damage Assessment
--*
```

```
--SEND-ON-CHANGE ONLY

Ownship_Damage_Occurrence :
   Moving_Model_Types.
   Scoring_Damage_Data;
--
-- Destination: ENV, EW, IOS, FD, PRO, FC, FS, NAV, RDR, PHC, VIS
--
Ownship_Scoring_Activation :
   Moving_Model_Types.
   Scoring_Activation_Status;
--
-- Destination: EW, IOS
--


--***************Function:
--*
--/ 10.13.6   Weapons Support
--*

-- SEND-ON-CHANGE OUTPUTS

Weapons_Discrete_Change :
   Weapons_Output_Interface_Types.
   Weapons_Discrete_Output_List;
--
-- Destination:  FS
--

Weapon_Deactivation :
   Moving_Model_Types.
   Moving_Model_Deactivation;
--
-- Destination: ENV, RDR, VIS, IOS, EW
--

Weapons_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
```

```
-- Destination : IOS
--

Jettison_Status_Change_Of_State :
   Global_Message_Types.
   Jettison_Status;
--
-- Destination : ENV, FS
--

HUD_Max_Rate_Output :
   Weapons_Output_Interface_Types.
   HUD_Output_Record;
--
-- Destination : FS
--

Stores_Config :
   Global_Message_Types.
   Weapon_Station_Change;
--
-- Destination : FS, IOS
--

Weapons_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Weapons_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Weapons_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
```

```
--
-- Destination : IOS
--

Weapons_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Weapons_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Weapons_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

--*
end Weapons_Output_Interface;
```

```
------------------------------------------------------------------------
-- %Z% Unit Name:         Radar_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:         Package Spec (no body)
-- %Z% Unit ID:           (tbd)
-- %Z% Author:            Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:          %I%
-- %Z% Delta Date:        %G%
-- %Z% Current Release: %R%
------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Radar segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
with Control_Types;
with Moving_Model_Types;
--
package Radar_Output_Interface_Types is
--
--***************************************************************
--*                                                             *
--/ 10.14   Radar Output Interface Types                        *
--*                                                             *
--***************************************************************
```

```
--*****************************************
--*
--/ 10.14.1   Aircraft/Simulator Specific Radar Types
--*
--*****************************************

-- These constants are important only if you collect targets or
-- moving models together into one message, rather than sending
-- one at a time.
Maximum_Number_Of_Air_To_Air_Targets    : constant := 10;
Maximum_Number_Of_Air_To_Ground_Targets : constant := 10;
Maximum_Number_Of_Targets               : constant := 10;
Maximum_Number_Of_Moving_Models         : constant := 10;


-- This is the count of the higher order harmonics of the
-- operating frequency.  Adapt as required for a specific
-- application.
Maximum_Number_Of_Frequency_Harmonics : constant := 5;

-- This and many of the following type declarations will be adapted
-- by deleting those modes, sensor types, etc. which are inactive
-- in this simulation.
type Specific_Mode_Type is (
   Ground_Map,
   Raid_Assessment,
   Search,
   Track,
   Track_While_Scan);

subtype PRF_Type is Base_Types.Unsigned_Integer_32 range 1..100000;

type Antenna_Index is (
   Main_Antenna,
   Flood_Horn);

type Assigning_Sensor_Type is (
   Tactical_Radar,
   SAR);
```

```
type Assigned_Sensor_Type is (
   FLIR,
   LLTV,
   LST,
   RWR);

type Sensor_Source is (
   Tactical_Radar,
   SAR,
   FLIR,
   LLTV,
   LST);

type Beam_Shape_Type is (
   Pencil,
   Fan,
   Flood);

type Waveform_Type is (
   Sawtooth,
   Sine,
   Square);

type Search_Pattern_Type is (
   Vertical_Scan,
   Horizontal_Scan,
   Boresight);

type Data_Update_Rate is (
   Request_High,
   Allow_Low);

-- Declare all D/Os in the flight station that this segment will
-- turn on or off
type Radar_Discrete_Outputs is
   (To_Be_Determined);

-- Characteristics which the radar's sensor fusion makes known to
-- EW or VIS on this simulator or radar system
type Sensor_Fused_Discernable_Characteristics (
   Sensor                        : Sensor_Source := Tactical_Radar;
```

```
   Relative_Velocity_Resolved : Base_Types.Sim_Boolean :=
Base_Types.False;
   Range_Resolved             : Base_Types.Sim_Boolean :=
Base_Types.False) is
 record
   Sensing_Device                          : Sensor_Source := Sensor;
  Moving_Model                        : Moving_Model_Types.Moving_Model_ID;
  Gimbal_Angles                       : Engineering_Units.Polar_Direction;
   Moving_Model_Cross_Section_Azimuth    : Engineering_Units.Radians;
   Moving_Model_Cross_Section_Elevation : Engineering_Units.Radians;
   case Sensor is
    when Flir =>
      Temperature                          : Engineering_Units.Degrees_C;
    when Tactical_Radar | SAR =>
      Range_Rate_Known                     : Base_Types.Sim_Boolean :=
                                             Relative_Velocity_Resolved;
      Range_Known                          : Base_Tvpes.Sim_Boolean :=
                                             Range_Resolved;
      case Relative_Velocity_Resolved is
       when Base_Types.True =>
         Range_Rate                        : Engineering_Units.Knots;
         case Range_Resolved is
          when Base_Types.True =>
          Slant_Range                  : Engineering_Units.Nautical_Miles;
          when Base_Types.False =>
            null;
         end case;
       when Base_Types.False =>
         null;
      end case;
    when Lltv | Lst =>
      null;
   end case;
end record;


--****************************************
--*
--/ 10.14.2   Aircraft/Simulator Reusable Radar Types
--*
--****************************************
```

```
-- Use this type if the simulation is sending data on one target at a
-- time (i.e., potentially several messages of this type in a frame)
type Air_To_Air_Target_Data is
 record
   Designated_Target_Location        :
Engineering_Units.Earth_Position_Components;
  Designated_Target_Identification : Moving_Model_Types.Moving_Model_ID;
   Designated_Target_Motion          :
Engineering_Units.Earth_Velocity_Components;
   Designated_Target_Attitude        :
Engineering_Units.Angular_Position_Components;
   Designated_Target_Tracking        : Base_Types.Sim_Boolean;
end record;

subtype Air_To_Air_Target_Count is Base_Types.Unsigned_Integer_32 range
   1..Maximum_Number_Of_Air_To_Air_Targets;

type Air_To_Air_Target_Array is
   array (Air_To_Air_Target_Count) of
   Air_To_Air_Target_Data;

-- Use this type if the simulation sends data on several targets at
-- the same time (i.e., one message per frame)
type Air_To_Air_Target_Data_Record is
 record
   Number_Of_Targets  : Air_To_Air_Target_Count;
   Air_To_Air_Targets : Air_To_Air_Target_Array;
end record;

-- Use this type if the simulation is sending data on one target at a
-- time (i.e., potentially several messages of this type in a frame)
type Target_Detection_State is
 record
  Designated_Target_Identification : Moving_Model_Types.Moving_Model_ID;
   Illuminated                      : Base_Types.Sim_Boolean;
   Elevation_And_Azimuth_Resolved   : Base_Types.Sim_Boolean;
   Relative_Velocity_Resolved       : Base_Types.Sim_Boolean;
   Acquisition_Attempted            : Base_Types.Sim_Boolean;
   Range_Resolved                   : Base_Types.Sim_Boolean;
   Lockon                           : Base_Types.Sim_Boolean;
end record;
```

```
subtype Target_Count is Base_Types.Unsigned_Integer_32 range
   1..Maximum_Number_Of_Targets;

type Target_Detection_State_Array is
   array (Target_Count) of
   Target_Detection_State;

-- Use this type if the simulation sends data on several targets at
-- the same time (i.e., one message per frame)
type Target_Detection_State_Record is
 record
   Number_Of_Targets     : Target_Count;
   Target_Detection_Data : Target_Detection_State_Array;
end record;

type Coarse_Mode_Type is (
   Active,
   Passive,
   Off);

subtype Frequency_Harmonics_Count is Base_Types.Unsigned_Integer_32
   range 1..Maximum_Number_Of_Frequency_Harmonics;

type Active_Freq_Array is
   array (Frequency_Harmonics_Count) of
   Base_Types.Float_32;

type Selected_Antenna_Type is
   array (Antenna_Index) of Base_Types.Discrete_State;

type Radar_Generic_Information is
 record
   Active_Frequency      : Active_Freq_Array;
   Center_Beam_Direction : Engineering_Units.Polar_Direction;
   Power_Factor          : Engineering_Units.Watts;
                           -- Computed as transmitter
                           -- power times transmit antenna gain.
   PRF                   : PRF_Type;
   Data_Validity_Time    : Control_Types.Frame_Number;
   Selected_Antenna      : Selected_Antenna_Type;
   Coarse_Mode           : Coarse_Mode_Type;
```

```
   Specific_Mode          : Specific_Mode_Type;
end record;

subtype Air_To_Ground_Target_Count is Base_Types.Unsigned_Integer_32
range
   1..Maximum_Number_Of_Air_To_Ground_Targets;

type Radar_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types. Radar_Hydraulic_Component)
   of Engineering_Units.Gal_Per_Min;

Number_Of_Radar_Discrete_Outputs : constant :=
   Radar_Discrete_Outputs'pos (Radar_Discrete_Outputs'last) -
   Radar_Discrete_Outputs'pos (Radar_Discrete_Outputs'first) + 1;


--*****************************************
--*
--/ 10.14.3   Radar Segment Output Records
--*
--*****************************************


--***************Function:
--*
--/ 10.14.3.1   Radar Processor
--*

-- NONE


--***************Function:
--*
--/ 10.14.3.2   Radar Image Generation
--*

type Image_Generation_Moving_Models_Half_Rate is
 record
   Current_Air_To_Air_Target_Data : Air_To_Air_Target_Data_Record;
--WPN
end record;
-- If multiple messages about target data are to be sent in a frame,
```

```
-- rather than collecting target data into one message which is sent
-- once a frame, then comment out the definition above and use the
-- two definitions below:
--
-- Send one of these
-- type Image_Generation_Moving_Models_Half_Rate is
--  record
--    Air_To_Air_Targets_Detected : Air_To_Air_Target_Count;  --WPN
-- end record;
--
-- And send 0 or more of these (where the number will match the
-- Air_To_Air_Targets_Detected field of the above message).
-- type Target_Data_Half_Rate is
--  record
--    Air_To_Air_Targets : Air_To_Air_Target_Data;  --WPN
-- end record;

type Image_Generation_Moving_Models_Quarter_Rate is
 record
   Target_Detection_Data : Target_Detection_State_Record;
   Radar_Generic_Data    : Radar_Generic_Information; --EW
end record;
-- If multiple messages about target data are to be sent in a frame,
-- rather than collecting target data into one message which is sent
-- once a frame, then comment out the definition above and use the
-- two definitions below:
--
-- Send one of these
-- type Image_Generation_Moving_Models_Quarter_Rate is
--  record
--    Targets_Detected    : Target_Count;
--    Radar_Generic_Data : Radar_Generic_Information; --EW
-- end record;
--
-- And send 0 or more of these (where the number will match the
-- Targets_Detected field of the above message).
-- type Target_Data_Quarter_Rate is
--  record
--    Target_Detection_Data : Target_Detection_State;
-- end record;
```

```
-- SEND-ON-CHANGE

type Rejected_Air_To_Air_Target_Array is
   array (Air_To_Air_Target_Count) of
   Moving_Model_Types.Moving_Model_ID;

type Air_To_Air_Reject_Data is
 record
   Number_Of_Rejected_AA_Targets : Air_To_Air_Target_Count;
   Rejected_AA_Targets           : Rejected_Air_To_Air_Target_Array;
end record;
-- If multiple messages about rejected targets are to be sent in a frame,
-- rather than collecting data into one message which is sent once a
-- frame, then comment out the two definitions above and use the
-- definitions below:
--
-- type Air_To_Air_Reject_Data is
--  record
--    Rejected_AA_Target : Moving_Model_Types.Moving_Model_ID;
-- end record;

type Air_To_Ground_Target_Array is
   array (Air_To_Ground_Target_Count) of
   Engineering_Units.Earth_Position_Components;

type Air_To_Ground_Target_Data is
 record
   Number_Of_AG_Targets : Air_To_Ground_Target_Count;
   Designated_Points    : Air_To_Ground_Target_Array;
end record;
-- If multiple messages about air-ground targets are to be sent in a
frame,
-- rather than collecting data into one message which is sent once a
-- frame, then comment out the two definitions above and use the
-- definition below:
--
-- type Air_To_Ground_Target_Data is
--  record
--    Designated_Point : Engineering_Units.Earth_Position_Components;
-- end record;

type Air_To_Ground_Reject_Data is
```

```
 record
   Number_Of_Rejected_AG_Targets : Air_To_Ground_Target_Count;
   Rejected_AG_Targets           : Air_To_Ground_Target_Array;
end record;
-- If multiple messages about rejected targets are to be sent in a frame,
-- rather than collecting data into one message which is sent once a
-- frame, then comment out the definition above and use the
-- definition below:
--
-- type Air_To_Ground_Reject_Data is
--  record
--    Rejected_AG_Target : Engineering_Units.Earth_Position_Components;
-- end record;

type Moving_Model_Data_Update_Rate_Change_Request is
 record
   ID          : Moving_Model_Types.Moving_Model_ID;
   Update_Rate : Data_Update_Rate;
end record;

type Ownship_Beacon_Data is
 record
   Beacon_Range : Engineering_Units.Nautical_Miles;
   Beacon_Code  : Base_Types.Unsigned_Integer_32;
end record;


--***************Function:
--*
--/ 10.14.3.3   Airborne Interrogate Sensor
--*

--SEND-ON-CHANGE

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Moving_Model_IFF_Data is
 record
   ID     : Moving_Model_Types.Moving_Model_ID;
```

```
   IFF_ID : Global_Message_Types.IFF_Data;
end record;

--
-- ...or it will collect the moving model data into an array;
--
type IFF_Interrogated_Model_Array is
   array (Moving_Model_Types.Moving_Model_Count) of
   Moving_Model_IFF_Data;


--
-- ...and send the ones which have changed in one of these messages:
--
type Airborne_IFF_Interrogate_Data is
 record
  Number_Of_Interrogated_Models : Moving_Model_Types.Moving_Model_Count;
   Moving_Model_IFF_Data        : IFF_Interrogated_Model_Array;
end record;


--***************Function:
--*
--/ 10.14.3.4   Terrain Following/Terrain Avoidance/
--/                  Radar Guidance
--*

--SEND-ON-CHANGE

type TF_TA_Data is
 record
   Terrain_Following_Pitch_Command : Engineering_Units.Radians;
   Radar_Roll_Command              : Engineering_Units.Radians;
   Radar_Yaw_Command               : Engineering_Units.Radians;
   Terrain_Following_Engaged       : Base_Types.Sim_Boolean;
   Terrain_Following_Valid         : Base_Types.Sim_Boolean;
   Radar_Guidance_Engaged          : Base_Types.Sim_Boolean;
   Radar_Guidance_Valid            : Base_Types.Sim_Boolean;
end record;


--***************Function:
```

```
--*
--/ 10.14.3.5   Mission Computer Interface
--*


type Mission_Computer_Interface_Half_Rate is
 record
   Location_Estimate_And_Flight_Regime : Air_To_Air_Target_Data; -- EW,
VIS
end record;

--SEND-ON-CHANGE

type Mission_Computer_Sensor_Assignment_Data is
 record
   Moving_Model      : Moving_Model_Types.Moving_Model_ID;
   Assigning_Sensor  : Assigning_Sensor_Type;
   Assigned_Sensor   : Assigned_Sensor_Type;
   Lost_Track        : Base_Types.Sim_Boolean;
end record;

-- See 10.14.3.1 above for definition of
-- Sensor_Fused_Discernable_Characteristics

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Single_Threat_Probability_Data is
 record
   Threat_ID                   : Moving_Model_Types.Moving_Model_ID;
   Probability_Of_That_Threat  : Engineering_Units.Normalized;
end record;

--
-- ...or it will collect the threats into an array;
--
type Threat_Probability_Array is
   array (Moving_Model_Types.Moving_Model_Count) of
   Single_Threat_Probability_Data;

--
```

```
-- ...and send the ones which have changed in one of these messages:
--
type Threat_Probability_Record is
 record
   Number_Of_Threats : Moving_Model_Types.Moving_Model_Count;
   Probable_Threats  : Threat_Probability_Array;
end record;

type Threat_Priority_Data is
 record
   Model_Number    : Moving_Model_Types.Moving_Model_ID;
   Threat_Priority : Engineering_Units.Normalized;
end record;

type Dynamic_Radar_Data is
 record
   Beam_Shape     : Beam_Shape_Type;
   Waveform       : Waveform_Type;
   Dwell_Time     : Engineering_Units.Seconds;
   Revisit_Time   : Engineering_Units.Seconds;
   Search_Pattern : Search_Pattern_Type;
end record;


--***************Function:
--*
--/ 10.14.3.6    Radar Aircraft Systems Interface
--*

type Radar_Aircraft_Systems_Interface_Eighth_Rate is
 record
   Electrical_Loads            : Global_Message_Types.
                                 Aircraft_Electrical_Bus_Load_Array;
                                 --FS
   Hydraulic_Component_Flows : Radar_Hydraulic_Component_Flow_Array;
                                 --FS
end record;


--***************Function:
--*
```

```
--/ 10.14.3.7    Crew Station Hardware Panel Interface
--*

--SEND-ON-CHANGE

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Radar_Discrete_Output_And_State is record
   ID    : Radar_Discrete_Outputs;
   State : Base_Types.Discrete_State;
 end record;

--
-- ...or it will collect the discretes into an array;
--
subtype Radar_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32
   range 1..Number_Of_Radar_Discrete_Outputs;

type Radar_Discrete_Output_Array is array (
   Radar_Discrete_Output_Count) of
   Radar_Discrete_Output_And_State;

--
-- ...and send the ones which have changed in one of these messages:
--
type Radar_Discrete_Output_List is
 record
   Number_Of_DOs     : Radar_Discrete_Output_Count;
   Discrete_Outputs : Radar_Discrete_Output_Array;
 end record;


--****************Function:
--*
--/ 10.14.3.8   Radar Database
--*

-- See Service_Function_Types
```

```
--***************Function:
--*
--/ 10.14.3.9   Visual Database
--*

-- See Service_Function_Types


--***************Function:
--*
--/ 10.14.3.10  Spatial Relations
--*

-- See Service_Function_Types


--***************Function:
--*
--/ 10.14.3.11  Occulting
--*

-- See Service_Function_Types


--***************Function:
--*
--/ 10.14.3.12  Radar Support
--*

-- SEND-ON-CHANGE ONLY

-- See Moving_Model_Types for definition of Emitter_Unique_Data

-- See Control_Types for responses to IOS

--*****************************************
--*
--/ 10.14.4    Radar Representation Specs
--*
```

```
--*************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes          : constant := 8; -- Bits per byte
   Byte_Size      : constant := 1 * Bytes;
   Halfword_Size  : constant := 2 * Bytes;
   Word_Size      : constant := 4 * Bytes;

   -- 10.14.3.2
   Moving_Model_ID_Size : constant :=
      Moving_Model_Types.Moving_Model_ID_Size;
   Earth_Position_Components_Size : constant :=
      Engineering_Units.Earth_Position_Components_Size;
   Earth_Velocity_Components_Size : constant :=
      Engineering_Units.Earth_Velocity_Components_Size;
   Angular_Position_Components_Size : constant :=
      Engineering_Units.Angular_Position_Components_Size;

   for Air_To_Air_Target_Data use
    record
      Designated_Target_Location
         at 0
         range 0..Earth_Position_Components_Size-1;
      Designated_Target_Identification
         at Earth_Position_Components_Size/Bytes
         range 0..Moving_Model_ID_Size-1;
      Designated_Target_Motion
         at Earth_Position_Components_Size/Bytes +
            Moving_Model_ID_Size/Bytes
         range 0..Earth_Velocity_Components_Size-1;
      Designated_Target_Attitude
         at Earth_Position_Components_Size/Bytes +
            Moving_Model_ID_Size/Bytes +
            Earth_Velocity_Components_Size/Bytes
         range 0..Angular_Position_Components_Size-1;
      Designated_Target_Tracking
         at Earth_Position_Components_Size/Bytes +
            Moving_Model_ID_Size/Bytes +
```

```
            Earth_Velocity_Components_Size/Bytes +
            Angular_Position_Components_Size/Bytes
         range 0..Byte_Size-1;
end record;
Air_To_Air_Target_Data_Size : constant :=
   Earth_Position_Components_Size +
   Moving_Model_ID_Size +
   Earth_Velocity_Components_Size +
   Angular_Position_Components_Size +
   Byte_Size;
for Air_To_Air_Target_Data'size use
   Air_To_Air_Target_Data_Size;

Air_To_Air_Target_Array_Size : constant :=
   Air_To_Air_Target_Data_Size *
   Maximum_Number_Of_Air_To_Air_Targets;

for Air_To_Air_Target_Array'size use
   Air_To_Air_Target_Array_Size;

for Air_To_Air_Target_Data_Record use
 record
   Number_Of_Targets
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Air_To_Air_Targets
     at 1 * Word_Size/Bytes range 0..Air_To_Air_Target_Array_Size-1;
end record;
Air_To_Air_Target_Data_Record_Size : constant :=
   Word_Size + Air_To_Air_Target_Array_Size;
for Air_To_Air_Target_Data_Record'size use
   Air_To_Air_Target_Data_Record_Size;

for Image_Generation_Moving_Models_Half_Rate'size use
   Air_To_Air_Target_Data_Record_Size;

for Target_Detection_State use
 record
   Designated_Target_Identification
      at 0 range 0..Moving_Model_ID_Size-1;
   Illuminated
      at Moving_Model_ID_Size/Bytes
```

```
      range 0..Byte_Size-1;
   Elevation_And_Azimuth_Resolved
      at Moving_Model_ID_Size/Bytes + 1 * Byte_Size/Bytes
      range 0..Byte_Size-1;
   Relative_Velocity_Resolved
      at Moving_Model_ID_Size/Bytes + 2 * Byte_Size/Bytes
      range 0..Byte_Size-1;
   Acquisition_Attempted
      at Moving_Model_ID_Size/Bytes + 3 * Byte_Size/Bytes
      range 0..Byte_Size-1;
   Range_Resolved
      at Moving_Model_ID_Size/Bytes + 4 * Byte_Size/Bytes
      range 0..Byte_Size-1;
   Lockon
      at Moving_Model_ID_Size/Bytes + 5 * Byte_Size/Bytes
      range 0..Byte_Size-1;
end record;
Target_Detection_State_Size : constant :=
   Moving_Model_ID_Size + 6 * Byte_Size;
for Target_Detection_State'size use
   Target_Detection_State_Size;

Target_Detection_State_Array_Size : constant :=
   Target_Detection_State_Size * Maximum_Number_Of_Targets;
for Target_Detection_State_Array'size use
   Target_Detection_State_Array_Size;

for Target_Detection_State_Record use
 record
   Number_Of_Targets
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Target_Detection_Data
     at 1 * Word_Size/Bytes range 0..Target_Detection_State_Array_Size-1;
end record;

Target_Detection_State_Record_Size : constant :=
   Word_Size + Target_Detection_State_Array_Size;
for Target_Detection_State_Record'size use
   Target_Detection_State_Record_Size;

Active_Freq_Array_Size : constant :=
```

```
   Word_Size * Maximum_Number_Of_Frequency_Harmonics;
for Active_Freq_Array'size use
   Active_Freq_Array_Size;

Polar_Direction_Size : constant :=
   Engineering_Units.Polar_Direction_Size;

Number_Of_Antennas : constant :=
   Antenna_Index'pos (
   Antenna_Index'last) -
   Antenna_Index'pos (
   Antenna_Index'first) + 1;

Selected_Antenna_Type_Size : constant :=
   Byte_Size * Number_Of_Antennas;
for Selected_Antenna_Type'size use
   Selected_Antenna_Type_Size;

Coarse_Mode_Type_Size : constant := Byte_Size;
for Coarse_Mode_Type'size use
   Coarse_Mode_Type_Size;

Specific_Mode_Type_Size : constant := Byte_Size;
for Specific_Mode_Type'size use
   Specific_Mode_Type_Size;

for Radar_Generic_Information use
 record
   Active_Frequency
      at 0
      range 0..Active_Freq_Array_Size-1;
   Center_Beam_Direction
      at Active_Freq_Array_Size/Bytes
      range 0..Polar_Direction_Size-1;
   Power_Factor
      at Active_Freq_Array_Size/Bytes +
         Polar_Direction_Size/Bytes
      range 0..Word_Size-1;
   PRF
      at Active_Freq_Array_Size/Bytes +
         Polar_Direction_Size/Bytes +
```

```
          Word_Size/Bytes
       range 0..Word_Size-1;
    Data_Validity_Time
       at Active_Freq_Array_Size/Bytes +
          Polar_Direction_Size/Bytes +
          2 * Word_Size/Bytes
       range 0..Word_Size-1;
    Selected_Antenna
       at Active_Freq_Array_Size/Bytes +
          Polar_Direction_Size/Bytes +
          3 * Word_Size/Bytes
       range 0..Selected_Antenna_Type_Size-1;
    Coarse_Mode
       at Active_Freq_Array_Size/Bytes +
          Polar_Direction_Size/Bytes +
          3 * Word_Size/Bytes +
          Selected_Antenna_Type_Size/Bytes
       range 0..Coarse_Mode_Type_Size-1;
    Specific_Mode
       at Active_Freq_Array_Size/Bytes +
          Polar_Direction_Size/Bytes +
          3 * Word_Size/Bytes +
          Selected_Antenna_Type_Size/Bytes +
          Coarse_Mode_Type_Size/Bytes
       range 0..Specific_Mode_Type_Size-1;
 end record;
 Radar_Generic_Information_Size : constant :=
    Active_Freq_Array_Size +
    Polar_Direction_Size +
    3 * Word_Size +
    Selected_Antenna_Type_Size +
    Coarse_Mode_Type_Size +
    Specific_Mode_Type_Size;

 for Radar_Generic_Information'size use
    Radar_Generic_Information_Size;

 for Image_Generation_Moving_Models_Quarter_Rate use
  record
    Target_Detection_Data
       at 0
```

```
            range 0..Target_Detection_State_Record_Size-1;
         Radar_Generic_Data
            at Target_Detection_State_Record_Size/Bytes
            range 0..Radar_Generic_Information_Size-1;
   end record;
   for Image_Generation_Moving_Models_Quarter_Rate'size use
      Target_Detection_State_Record_Size +
      Radar_Generic_Information_Size;

   Rejected_Air_To_Air_Target_Array_Size : constant :=
      Moving_Model_ID_Size *
      Maximum_Number_Of_Air_To_Air_Targets;
   for Rejected_Air_To_Air_Target_Array'size use
      Rejected_Air_To_Air_Target_Array_Size;

   for Air_To_Air_Reject_Data use
    record
      Number_Of_Rejected_AA_Targets
         at 0 range 0..Word_Size-1;
      Rejected_AA_Targets
       at Word_Size/Bytes range 0..Rejected_Air_To_Air_Target_Array_Size-1;
   end record;
   for Air_To_Air_Reject_Data'size use
      Word_Size +
      Rejected_Air_To_Air_Target_Array_Size;

   Air_To_Ground_Target_Array_Size : constant :=
      Earth_Position_Components_Size *
      Maximum_Number_Of_Air_To_Ground_Targets;
   for Air_To_Ground_Target_Array'size use
      Air_To_Ground_Target_Array_Size;

   for Air_To_Ground_Target_Data use
    record
      Number_Of_AG_Targets
         at 0 range 0..Word_Size-1;
      Designated_Points
         at Word_Size/Bytes range 0..Air_To_Ground_Target_Array_Size-1;
   end record;
   for Air_To_Ground_Target_Data'size use
      Word_Size +
```

```
   Air_To_Ground_Target_Array_Size;

for Air_To_Ground_Reject_Data use
 record
   Number_Of_Rejected_AG_Targets
      at 0 range 0..Word_Size-1;
   Rejected_AG_Targets
      at Word_Size/Bytes range 0..Air_To_Ground_Target_Array_Size-1;
end record;
for Air_To_Ground_Reject_Data'size use
   Word_Size +
   Air_To_Ground_Target_Array_Size;

Data_Update_Rate_Size : constant := Byte_Size;
for Data_Update_Rate'size use Data_Update_Rate_Size;

for Moving_Model_Data_Update_Rate_Change_Request use
 record
   ID
      at 0 range 0..Moving_Model_ID_Size-1;
   Update_Rate
      at Moving_Model_ID_Size/Bytes
      range 0..Data_Update_Rate_Size-1;
end record;
for Moving_Model_Data_Update_Rate_Change_Request'size use
   Moving_Model_ID_Size + Data_Update_Rate_Size;

for Ownship_Beacon_Data use
 record
   Beacon_Range
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
   Beacon_Code
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
end record;
for Ownship_Beacon_Data'size use 2 * Word_Size;

-- 10.14.3.3
IFF_Data_Size : constant :=
   Global_Message_Types.IFF_Data_Size;

for Moving_Model_IFF_Data use
```

```
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   IFF_ID
      at Moving_Model_ID_Size/Bytes
      range 0..IFF_Data_Size-1;
end record;
Moving_Model_IFF_Data_Size : constant :=
   Moving_Model_ID_Size + IFF_Data_Size;
for Moving_Model_IFF_Data'size use Moving_Model_IFF_Data_Size;

IFF_Interrogated_Model_Array_Size : constant :=
   Moving_Model_IFF_Data_Size *
   Moving_Model_Types.Maximum_Number_Of_Moving_Models;
for IFF_Interrogated_Model_Array'size use
   IFF_Interrogated_Model_Array_Size;

for Airborne_IFF_Interrogate_Data use
 record
   Number_Of_Interrogated_Models
      at 0 range 0..Word_Size-1;
   Moving_Model_IFF_Data
      at Word_Size/Bytes
      range 0..IFF_Interrogated_Model_Array_Size-1;
end record;
for Airborne_IFF_Interrogate_Data'size use
   Word_Size +IFF_Interrogated_Model_Array_Size;

-- 10.14.3.4
for TF_TA_Data use
 record
   Terrain_Following_Pitch_Command
      at 0 * Word_Size/Bytes
      range 0..Word_Size-1;
   Radar_Roll_Command
      at 1 * Word_Size/Bytes
      range 0..Word_Size-1;
   Radar_Yaw_Command
      at 2 * Word_Size/Bytes
      range 0..Word_Size-1;
```

```
   Terrain_Following_Engaged
      at 3 * Word_Size/Bytes
      range 0..Byte_Size-1;
   Terrain_Following_Valid
      at 3 * Word_Size/Bytes + 1 * Byte_Size/Bytes
      range 0..Byte_Size-1;
   Radar_Guidance_Engaged
      at 3 * Word_Size/Bytes + 2 * Byte_Size/Bytes
      range 0..Byte_Size-1;
   Radar_Guidance_Valid
      at 3 * Word_Size/Bytes + 3 * Byte_Size/Bytes
      range 0..Byte_Size-1;
end record;
for TF_TA_Data'size use 3 * Word_Size + 4 * Byte_Size;

-- 10.14.3.5
for Mission_Computer_Interface_Half_Rate'size use
   Air_To_Air_Target_Data_Size;

Assigning_Sensor_Type_Size : constant := 8;
for Assigning_Sensor_Type'size use
   Assigning_Sensor_Type_Size;

Assigned_Sensor_Type_Size : constant := 8;
for Assigned_Sensor_Type'size use
   Assigned_Sensor_Type_Size;

for Mission_Computer_Sensor_Assignment_Data use
 record
   Moving_Model
      at 0
      range 0..Moving_Model_ID_Size-1;
   Assigning_Sensor
      at Moving_Model_ID_Size/Bytes
      range 0..Assigning_Sensor_Type_Size-1;
   Assigned_Sensor
      at Moving_Model_ID_Size/Bytes +
         Assigning_Sensor_Type_Size/Bytes
      range 0..Assigned_Sensor_Type_Size-1;
   Lost_Track
      at Moving_Model_ID_Size/Bytes +
```

```
          Assigning_Sensor_Type_Size/Bytes +
          Assigned_Sensor_Type_Size/Bytes
       range 0..Byte_Size-1;
end record;
for Mission_Computer_Sensor_Assignment_Data'size use
   Moving_Model_ID_Size +
   Assigning_Sensor_Type_Size +
   Assigned_Sensor_Type_Size +
   Byte_Size;

Sensor_Source_Size : constant := Halfword_Size;
for Sensor_Source'size use Sensor_Source_Size;

for Sensor_Fused_Discernable_Characteristics use
 record
   Sensor
      at 0
      range 0..Sensor_Source_Size-1;
   Relative_Velocity_Resolved
      at Sensor_Source_Size/Bytes
      range 0..Byte_Size-1;
   Range_Resolved
      at Sensor_Source_Size/Bytes +
         Byte_Size/Bytes
      range 0..Byte_Size-1;
   Sensing_Device
      at Sensor_Source_Size/Bytes +
         Byte_Size/Bytes +
         Byte_Size/Bytes
      range 0..Sensor_Source_Size-1;
   -- 2 bytes spare
   Moving_Model
      at Sensor_Source_Size/Bytes +
         Byte_Size/Bytes +
         Byte_Size/Bytes +
         Sensor_Source_Size/Bytes +
         Halfword_Size/Bytes
      range 0..Moving_Model_ID_Size-1;
   Gimbal_Angles
      at Sensor_Source_Size/Bytes +
         Byte_Size/Bytes +
```

```
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes
   range 0..Polar_Direction_Size-1;
Moving_Model_Cross_Section_Azimuth
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes
   range 0..Word_Size-1;
Moving_Model_Cross_Section_Elevation
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes +
      Word_Size/Bytes
   range 0..Word_Size-1;
Temperature
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes
   range 0..Word_Size-1;
Range_Rate_Known -- same location as Temperature
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
```

```
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes
   range 0..Byte_Size-1;
Range_Known
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Byte_Size/Bytes
   range 0..Byte_Size-1;
-- 2 bytes spare
Range_Rate
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes +
      Word_Size/Bytes +
      Word_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Halfword_Size/Bytes
   range 0..Word_Size-1;
Slant_Range
   at Sensor_Source_Size/Bytes +
      Byte_Size/Bytes +
      Byte_Size/Bytes +
      Sensor_Source_Size/Bytes +
      Halfword_Size/Bytes +
      Moving_Model_ID_Size/Bytes +
      Polar_Direction_Size/Bytes +
      Word_Size/Bytes +
```

```
            Word_Size/Bytes +
            Byte_Size/Bytes +
            Byte_Size/Bytes +
            Halfword_Size/Bytes +
            Word_Size/Bytes
         range 0..Word_Size-1;
end record;
for Sensor_Fused_Discernable_Characteristics'size use
   Sensor_Source_Size +
   Byte_Size +
   Byte_Size +
   Sensor_Source_Size +
   Halfword_Size +           -- pad
   Moving_Model_ID_Size +
   Polar_Direction_Size +
   Word_Size +
   Word_Size +
   Byte_Size +
   Byte_Size +
   Halfword_Size +           -- pad
   Word_Size +
   Word_Size;

for Single_Threat_Probability_Data use
 record
   Threat_ID
      at 0 range 0..Moving_Model_ID_Size-1;
   Probability_Of_That_Threat
      at Moving_Model_ID_Size/Bytes range 0..Word_Size-1;
end record;
Single_Threat_Probability_Data_Size : constant :=
   Moving_Model_ID_Size + Word_Size;

for Single_Threat_Probability_Data'size use
   Single_Threat_Probability_Data_Size;

Threat_Probability_Array_Size : constant :=
   Single_Threat_Probability_Data_Size *
   Moving_Model_Types.Maximum_Number_Of_Moving_Models;
for Threat_Probability_Array'size use
   Threat_Probability_Array_Size;
```

```
for Threat_Probability_Record use
 record
   Number_Of_Threats
      at 0
      range 0..Word_Size-1;
   Probable_Threats
      at Word_Size/Bytes
      range 0..Threat_Probability_Array_Size-1;
end record;
for Threat_Probability_Record'size use
   Word_Size +
   Threat_Probability_Array_Size;

for Threat_Priority_Data use
 record
   Model_Number
      at 0
      range 0..Moving_Model_ID_Size-1;
   Threat_Priority
      at Moving_Model_ID_Size/Bytes
      range 0..Word_Size-1;
end record;
for Threat_Priority_Data'size use
   Moving_Model_ID_Size +
   Word_Size;

Beam_Shape_Type_Size : constant := Halfword_Size;
for Beam_Shape_Type'size use
   Beam_Shape_Type_Size;

Waveform_Type_Size : constant := Halfword_Size;
for Waveform_Type'size use
   Waveform_Type_Size;

Search_Pattern_Type_Size : constant := Byte_Size;
for Search_Pattern_Type'size use
   Search_Pattern_Type_Size;

for Dynamic_Radar_Data use
 record
```

```
   Beam_Shape
      at 0 range 0..Beam_Shape_Type_Size-1;
   Waveform
      at Beam_Shape_Type_Size/Bytes
      range 0..Waveform_Type_Size-1;
   Dwell_Time
      at Beam_Shape_Type_Size/Bytes +
         Waveform_Type_Size/Bytes
      range 0..Word_Size-1;
   Revisit_Time
      at Beam_Shape_Type_Size/Bytes +
         Waveform_Type_Size/Bytes +
         Word_Size/Bytes
      range 0..Word_Size-1;
   Search_Pattern
      at Beam_Shape_Type_Size/Bytes +
         Waveform_Type_Size/Bytes +
         Word_Size/Bytes +
         Word_Size/Bytes
      range 0..Search_Pattern_Type_Size-1;
end record;
for Dynamic_Radar_Data'size use
   Beam_Shape_Type_Size +
   Waveform_Type_Size +
   Word_Size +
   Word_Size +
   Search_Pattern_Type_Size;

-- 10.14.3.6
Aircraft_Electrical_Bus_Load_Array_Size : constant :=
   Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size;

Number_Of_Radar_Hydraulic_Components : constant :=
   Global_Message_Types.Radar_Hydraulic_Component'pos (
   Global_Message_Types.Radar_Hydraulic_Component'last) -
   Global_Message_Types.Radar_Hydraulic_Component'pos (
   Global_Message_Types.Radar_Hydraulic_Component'first) + 1;

Radar_Hydraulic_Component_Flow_Array_Size : constant :=
   Word_Size * Number_Of_Radar_Hydraulic_Components;
for Radar_Hydraulic_Component_Flow_Array'size use
```

```
      Radar_Hydraulic_Component_Flow_Array_Size;

   for Radar_Aircraft_Systems_Interface_Eighth_Rate use
    record
      Electrical_Loads
         at 0 range 0..Aircraft_Electrical_Bus_Load_Array_Size-1;
      Hydraulic_Component_Flows
         at Aircraft_Electrical_Bus_Load_Array_Size/Bytes
         range 0..Radar_Hydraulic_Component_Flow_Array_Size-1;
   end record;
   for Radar_Aircraft_Systems_Interface_Eighth_Rate'size use
      Aircraft_Electrical_Bus_Load_Array_Size +
      Radar_Hydraulic_Component_Flow_Array_Size;

   -- 10.14.3.7
   Radar_Discrete_Output_And_State_Size : constant := 2 * Bytes;
   for Radar_Discrete_Output_And_State'size use
      Radar_Discrete_Output_And_State_Size;

   Radar_Discrete_Output_Array_Size : constant :=
      Radar_Discrete_Output_And_State_Size *
      Number_Of_Radar_Discrete_Outputs;
   for Radar_Discrete_Output_Array'size use
      Radar_Discrete_Output_Array_Size;

   for Radar_Discrete_Output_List use
    record
      Number_Of_DOs    at 0
                       range 0..Word_Size-1;
      Discrete_Outputs at Word_Size/Bytes
                       range 0..
                       Radar_Discrete_Output_Array_Size-1;
    end record;
   for Radar_Discrete_Output_List'size use
      Word_Size + Radar_Discrete_Output_Array_Size;

--*
end Radar_Output_Interface_Types;
```

```
---------------------------------------------------------------------
-- %Z% Unit Name:         Radar_Output_Interface
-- %Z% Source Pathname:   %P%
-- %Z% Unit Type:         Package Spec (no body)
-- %Z% Unit ID:           (tbd)
-- %Z% Author:            Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:    12 August 1993
-- %Z% SCCS Filename:     %M%
-- %Z% Delta ID:          %I%
-- %Z% Delta Date:        %G%
-- %Z% Current Release:   %R%
---------------------------------------------------------------------
--
-- Purpose:
--   This package specifies all the message objects which are sent by the
--   Radar segment.
--
-- Adaptation:
--   The first step in adaptation is to determine which of the functions
--   in this segment will not be performed, based on simulator
requirements.
--   The messages associated with these functions need not be sent, and
--   should therefore be deleted or commented out.
--
--   Each message declaration is followed by a comment line containing
--   "Destination:" and the abbreviations of the segment(s) which receive
--   this message.  These comments should be modified to account for
--   (a) the presence or absence of other segments, and (b) the
requirements
--  of the other segments for data.  For example, if segment X is absent,
--   then the notation that segment X is a destination of a given
--   message should be removed.  Similarly, when segment Y does not
require
--   the data in a given message, then the notation that segment Y is a
--   destination for that message should be removed.
--
--   When the segment abbreviations have all been removed for a message,
--   it is clear that this message need not be sent, and the message
--   object declaration itself may be commented out or deleted.
--
--   The four service functions: Radar Database, Visual Database, Spatial
--   Relations and Occulting have messages which must each be sent by one
```

```
-- and only one segment.  Only one of the following three segments:
-- Environment, Radar or Visual, may send these messages.  Modify the
-- output interface packages for each of these three segments in
-- accordance with the assignment of the functions to the segments,
-- commenting or uncommenting declarations accordingly.
--
with Radar_Output_Interface_Types;
with Control_Types;
with Service_Function_Types;
with Moving_Model_Types;
--
package Radar_Output_Interface is
--
--*************************************************************
--*                                                           *
--/ 10.15   Radar Output Interface                            *
--*                                                           *
--*************************************************************


--***************Function:
--*
--/ 10.15.1   Radar Processor
--*

-- NONE


--***************Function:
--*
--/ 10.15.2   Radar Image Generation
--*

Image_Generation_Moving_Models_Half_Rate_Outputs :
   Radar_Output_Interface_Types.
   Image_Generation_Moving_Models_Half_Rate;
--
-- Destination: WPN
--

Image_Generation_Moving_Models_Quarter_Rate_Outputs :
```

```
   Radar_Output_Interface_Types.
   Image_Generation_Moving_Models_Quarter_Rate;
--
-- Destination: EW
--


--SEND-ON-CHANGE OUTPUTS

Air_To_Air_Reject_Data_Change :
   Radar_Output_Interface_Types.
   Air_To_Air_Reject_Data;
--
-- Destination: WPN
--

Designated_Air_To_Ground_Data_Change :
   Radar_Output_Interface_Types.
   Air_To_Ground_Target_Data;
--
-- Destination: WPN
--

Air_To_Ground_Reject_Data_Change :
   Radar_Output_Interface_Types.
   Air_To_Ground_Reject_Data;
--
-- Destination: WPN
--

Moving_Model_Data_Update_Rate_Change :
   Radar_Output_Interface_Types.
   Moving_Model_Data_Update_Rate_Change_Request;
--
-- Destination: ENV, EW, WPN, FD, VIS
--

Ownship_Beacon_Data_Change :
   Radar_Output_Interface_Types.
   Ownship_Beacon_Data;
--
```

```
-- Destination: ENV, EW, FD, IOS
--


--***************Function:
--*
--/ 10.15.3   Airborne Interrogate Sensor
--*

--SEND-ON-CHANGE OUTPUTS

IFF_Airborne_Interrogate_Data_Change :
   Radar_Output_Interface_Types.
   Airborne_IFF_Interrogate_Data;
--
-- Destination: NAV
--


--***************Function:
--*
--/ 10.15.4   Terrain Following/Terrain Avoidance/
--/               Radar Guidance
--*

--SEND-ON-CHANGE OUTPUTS

TF_TA_Data_Change :
   Radar_Output_Interface_Types.
   TF_TA_Data;
--
-- Destination: FC
--


--***************Function:
--*
--/ 10.15.5   Mission Computer Interface
--*

Mission_Computer_Interface_Half_Rate_Outputs :
```

```
    Radar_Output_Interface_Types.
    Mission_Computer_Interface_Half_Rate;
--
-- Destination: EW, VIS
--


--SEND-ON-CHANGE OUTPUTS

Mission_Computer_Sensor_Data_Change :
    Radar_Output_Interface_Types.
    Mission_Computer_Sensor_Assignment_Data;
--
-- Destination: EW, VIS
--

Sensor_Fused_Discernable_Characteristics_Change :
    Radar_Output_Interface_Types.
    Sensor_Fused_Discernable_Characteristics;
--
-- Destination: EW, VIS
--

Threat_Probability_Data_Change :
    Radar_Output_Interface_Types.
    Threat_Probability_Record;
--
-- Destination: EW, VIS
--

Threat_Priority_Data_Change :
    Radar_Output_Interface_Types.
    Threat_Priority_Data;
--
-- Destination: EW, VIS
--

Dynamic_Radar_Data_Change :
    Radar_Output_Interface_Types.
    Dynamic_Radar_Data;
--
```

```
-- Destination: EW
--


--***************Function:
--*
--/ 10.15.6   Radar Aircraft Systems Interface
--*

Radar_Aircraft_Systems_Interface_Eighth_Rate_Outputs :
   Radar_Output_Interface_Types.
   Radar_Aircraft_Systems_Interface_Eighth_Rate;
--
-- Destination: FS
--


--***************Function:
--*
--/ 10.15.7    Crew Station Hardware Panel Interface
--*

-- SEND-ON-CHANGE OUTPUTS

Radar_Discrete_Data_Output_Change :
   Radar_Output_Interface_Types.
   Radar_Discrete_Output_List;
--
-- Destination: FS
--

--***************Function:
--*
--/ 10.15.8    Radar Database
--*

-- NONE


--***************Function:
--*
```

```
--/ 10.15.9    Visual Database
--*

-- NONE


--***************Function:
--*
--/ 10.15.10  Spatial Relations
--*

--SEND-ON-CHANGE OUTPUTS

Position_Range_Change :
   Service_Function_Types.
   Position_Range_Update;
--
-- Destination: ENV, NAV, VIS
--

Groundspeed_Change :
   Service_Function_Types.
   Groundspeed_Update;
--
-- Destination: ENV, NAV, VIS
--


-- Ownship_Height_Above_Terrain_Max_Rate_Outputs :
--    Service_Function_Types.
--    Ownship_Height_Above_Terrain;
--
-- Destination: ENV, NAV, WPN, PHC, VIS, FD
-- Assigned to ENV

-- Moving_Models_Height_Above_Terrain_Max_Rate_Outputs :
--    Service_Function_Types.
--    Moving_Models_Height_Above_Terrain;
--
-- Destination: ENV, NAV, WPN, PHC, VIS, FD
-- Assigned to ENV
```

```
--***************Function:
--*
--/ 10.15.11  Occulting

--SEND-ON-CHANGE OUTPUTS

-- Occulting_Status_Change :
--    Service_Function_Types.
--    Occulting_Status_Update;
--
-- Destination: VIS, NAV, EW, ENV
-- Assigned to VIS


--***************Function:
--*
--/ 10.15.12  Radar Support
--*

-- SEND-ON-CHANGE OUTPUTS

Radar_Emitter_Unique_Data :
   Moving_Model_Types.
   Emitter_Unique_Data;
--
-- Destination: ENV
--

Radar_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Radar_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
```

```
--

Radar_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Radar_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Radar_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Radar_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Radar_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--

--*
end Radar_Output_Interface;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Electronic_Warfare_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Electronic_Warfare segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
with Moving_Model_Types;
--
package Electronic_Warfare_Output_Interface_Types is
--
--***************************************************************
--*                                                             *
--/ 10.16   Electronic Warfare Output Interface Types           *
--*                                                             *
--***************************************************************
```

```
--*****************************************
--*
--/ 10.16.1   Aircraft/Simulator Specific Electronic
--/              Warfare Types
--*
--*****************************************

-- This constant is important only if you send dynamic data on all
-- decoys at the same time, instead of sending them one per message
Maximum_Number_Of_Decoys : constant := 10;

-- Declare all D/Os in the flight station that this segment will
-- turn on or off
type Electronic_Warfare_Discrete_Outputs is
     (To_Be_Determined);


--*****************************************
--*
--/ 10.16.2   Aircraft/Simulator Reusable Electronic Warfare Types
--*
--*****************************************

type EW_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types.Electronic_Warfare_Hydraulic_Component)
   of Engineering_Units.Gal_Per_Min;

Number_Of_Electronic_Warfare_Discrete_Outputs : constant :=
   Electronic_Warfare_Discrete_Outputs'pos (
   Electronic_Warfare_Discrete_Outputs'last) -
   Electronic_Warfare_Discrete_Outputs'pos (
   Electronic_Warfare_Discrete_Outputs'first) + 1;

subtype Decoy_Count is Base_Types.Unsigned_Integer_32 range
   1..Maximum_Number_Of_Decoys;

type Decoy_Moving_Model_Dynamic_Data_Array is array (Decoy_Count)
   of Moving_Model_Types.Moving_Model_Dynamic_Data;

type Decoy_Moving_Model_Dynamic_Data_Record is
 record
```

```
   Number_Of_Decoys : Decoy_Count;
   Decoys           : Decoy_Moving_Model_Dynamic_Data_Array;
 end record;


--*****************************************
--*
--/ 10.16.3   Electronic Warfare Segment Output Records
--*
--*****************************************


--***************Function:
--*
--/ 10.16.3.1   Ownship Chaff and Flares
--*

-- see Moving_Model_Types for declarations of
-- Ownship_Chaff_And_Flares_Half_Rate_Outputs and
-- Ownship_Chaff_And_Flares_Sixteenth_Rate_Outputs


--***************Function:
--*
--/ 10.16.3.2   Dedicated Displays
--*

--NONE


--***************Function:
--*
--/ 10.16.3.3   Ownship Electronic Counter Measures (ECM)
--*

type Ownship_ECM_Half_Rate is
 record
   Decoy_Dynamic_Data : Decoy_Moving_Model_Dynamic_Data_Record;
                                -- RDR, NAV, IOS, VIS
 end record;
```

```
--
-- An alternative is to send data on one decoy at a time, in which
-- case, the declaration above will be:
--
-- type Ownship_ECM_Half_Rate is
--  record
--   Decoy_Dynamic_Data : Moving_Model_Types.Moving_Model_Dynamic_Data;
--                            -- RDR, NAV, IOS, VIS
-- end record;

-- see Moving_Model_Types for declaration of Decoy_Unique_Data


--***************Function:
--*
--/ 10.16.3.4   Pods and Controls
--*

type Pods_And_Controls_Eighth_Rate is
 record
   Electrical_Loads            : Global_Message_Types.
                                 Aircraft_Electrical_Bus_Load_Array;
                                 --FS
   Hydraulic_Component_Flows : EW_Hydraulic_Component_Flow_Array;
                                 --FS
 end record;


--***************Function:
--*
--/ 10.16.3.5   Radar Warning Receiver
--*

--NONE


--***************Function:
--*
--/ 10.16.3.6   Threat Detection
--*
```

```
--NONE

--****************Function:
--*
--/ 10.16.3.7   Electronic Warfare Support
--*

-- See Control_Types for responses to IOS

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Electronic_Warfare_Discrete_Output_And_State is record
   ID    : Electronic_Warfare_Discrete_Outputs;
   State : Base_Types.Discrete_State;
 end record;

--
-- ...or it will collect the discretes into an array;
--
subtype Electronic_Warfare_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32
   range 1..Number_Of_Electronic_Warfare_Discrete_Outputs;

type Electronic_Warfare_Discrete_Output_Array is array (
   Electronic_Warfare_Discrete_Output_Count) of
   Electronic_Warfare_Discrete_Output_And_State;

--
-- ...and send the ones which have changed in one of these messages:
--
type Electronic_Warfare_Discrete_Output_List is
 record
   Number_Of_DOs     : Electronic_Warfare_Discrete_Output_Count;
   Discrete_Outputs  : Electronic_Warfare_Discrete_Output_Array;
 end record;

--*****************************************
```

```
--*
--/ 10.16.4   Electronic Warfare Representation Specs
--*
--*****************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes     : constant := 8; -- Bits per byte
   Byte_Size : constant := 1 * Bytes;
   Word_Size : constant := 4 * Bytes;

   -- 10.16.3.3
   Decoy_Moving_Model_Dynamic_Data_Array_Size : constant :=
      Moving_Model_Types.Moving_Model_Dynamic_Data_Size *
      Maximum_Number_Of_Decoys;
   for Decoy_Moving_Model_Dynamic_Data_Array'size use
      Decoy_Moving_Model_Dynamic_Data_Array_Size;

   for Decoy_Moving_Model_Dynamic_Data_Record use
    record
      Number_Of_Decoys at 0 range 0..Word_Size-1;
      Decoys           at Word_Size/Bytes
                   range 0..Decoy_Moving_Model_Dynamic_Data_Array_Size-1;
   end record;

   Decoy_Moving_Model_Dynamic_Data_Record_Size : constant :=
      Word_Size +
      Decoy_Moving_Model_Dynamic_Data_Array_Size;
   for Decoy_Moving_Model_Dynamic_Data_Record'size use
      Decoy_Moving_Model_Dynamic_Data_Record_Size;

   for Ownship_ECM_Half_Rate'size use
      Decoy_Moving_Model_Dynamic_Data_Record_Size;
   -- When one decoy is sent at a time, use the following
   -- instead of the rep spec above:
   --
   -- for Ownship_ECM_Half_Rate'size use
   --    Moving_Model_Types.Moving_Model_Dynamic_Data_Size;
```

```
-- 10.16.3.4
Number_Of_Electronic_Warfare_Hydraulic_Components : constant :=
   Global_Message_Types.Electronic_Warfare_Hydraulic_Component'pos (
   Global_Message_Types.Electronic_Warfare_Hydraulic_Component'last)
-
   Global_Message_Types.Electronic_Warfare_Hydraulic_Component'pos (

Global_Message_Types.Electronic_Warfare_Hydraulic_Component'first) + 1;

   EW_Hydraulic_Component_Flow_Array_Size : constant :=
      Number_Of_Electronic_Warfare_Hydraulic_Components * Word_Size;
   for EW_Hydraulic_Component_Flow_Array'size use
      EW_Hydraulic_Component_Flow_Array_Size;

   for Pods_And_Controls_Eighth_Rate use
    record
      Electrical_Loads          at 0
                                range 0..Global_Message_Types.
                            Aircraft_Electrical_Bus_Load_Array_Size-1;
      Hydraulic_Component_Flows at Global_Message_Types.
                         Aircraft_Electrical_Bus_Load_Array_Size/Bytes
                         range 0..EW_Hydraulic_Component_Flow_Array_Size-1;
    end record;
   for Pods_And_Controls_Eighth_Rate'size use
      Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size +
      EW_Hydraulic_Component_Flow_Array_Size;

   -- 10.16.3.7
   Electronic_Warfare_Discrete_Output_And_State_Size : constant := 2 *
Bytes;
   for Electronic_Warfare_Discrete_Output_And_State'size use
      Electronic_Warfare_Discrete_Output_And_State_Size;

   Electronic_Warfare_Discrete_Output_Array_Size : constant :=
      Electronic_Warfare_Discrete_Output_And_State_Size *
      Number_Of_Electronic_Warfare_Discrete_Outputs;
   for Electronic_Warfare_Discrete_Output_Array'size use
      Electronic_Warfare_Discrete_Output_Array_Size;

   for Electronic_Warfare_Discrete_Output_List use
    record
      Number_Of_DOs     at 0
```

```
                          range 0..Word_Size-1;
      Discrete_Outputs at Word_Size/Bytes
                          range 0..
                          Electronic_Warfare_Discrete_Output_Array_Size-1;
    end record;
   for Electronic_Warfare_Discrete_Output_List'size use
      Word_Size + Electronic_Warfare_Discrete_Output_Array_Size;

--*
end Electronic_Warfare_Output_Interface_Types;
```

```
-------------------------------------------------------------------------
-- %Z% Unit Name:        Electronic_Warfare_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Electronic_Warfare segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
-- of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
with Electronic_Warfare_Output_Interface_Types;
with Control_Types;
```

```
with Moving_Model_Types;
--
package Electronic_Warfare_Output_Interface is
--
--*************************************************************
--*                                                           *
--/ 10.17   Electronic Warfare Output Interface               *
--*                                                           *
--*************************************************************


--***************Function:
--*
--/ 10.17.1   Ownship Chaff and Flares
--*

Ownship_Chaff_And_Flares_Half_Rate_Outputs :
   Moving_Model_Types.
   Chaff_And_Flares_Moving_Model_Data;
--
-- Destination:  ENV, RDR, VIS, IOS
--

Ownship_Chaff_And_Flares_Sixteenth_Rate_Outputs :
   Moving_Model_Types.
   Chaff_And_Flares_Detail_Data;
--
-- Destination:  ENV, RDR, VIS, IOS
--


--***************Function:
--*
--/ 10.17.2   Dedicated Displays
--*

--NONE


--***************Function:
--*
```

```
--/ 10.17.3   Ownship Electronic Counter Measures (ECM)
--*

Ownship_ECM_Half_Rate_Outputs :
   Electronic_Warfare_Output_Interface_Types.
   Ownship_ECM_Half_Rate;
--
-- Destination: ENV, RDR, VIS, IOS, NAV
--

--SEND-ON-CHANGE OUTPUTS

Decoy_Unique_Data :
   Moving_Model_Types.
   Decoy_Moving_Model_Unique_Data;
--
-- Destination: ENV, RDR, VIS, IOS, NAV
--


--***************Function:
--*
--/ 10.17.4   Pods and Controls
--*

Pods_And_Controls_Eighth_Rate_Outputs :
   Electronic_Warfare_Output_Interface_Types.
   Pods_And_Controls_Eighth_Rate;
--
-- Destination:  FS
--

--*NOTE: Emitter and IFF data is sent as part of
--       Platform_Moving_Model_Unique_Data from the "Threat
--       Platform Dynamics" function.


--***************Function:
--*
--/ 10.17.5   Radar Warning Receiver
--*
```

```
--NONE


--***************Function:
--*
--/ 10.17.6   Threat Detection
--*

--NONE


--***************Function:
--*
--/ 10.17.7 Electronic Warfare Support
--*

-- SEND-ON-CHANGE OUTPUTS

Electronic_Warfare_Discrete_Output_Change :
   Electronic_Warfare_Output_Interface_Types.
   Electronic_Warfare_Discrete_Output_List;
--
-- Destination:  FS
--

Electronic_Warfare_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Electronic_Warfare_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode  esponse;
--
-- Destination : IOS
--

Electronic_Warfare_Performance_Test_Response :
```

```
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Electronic_Warfare_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Electronic_Warfare_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Electronic_Warfare_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Electronic_Warfare_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--

--*
end Electronic_Warfare_Output_Interface;
```

```
-------------------------------------------------------------------
-- %Z% Unit Name:       Physical_Cues_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  3 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
-------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Physical_Cues segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
package Physical_Cues_Output_Interface_Types is
--
--**************************************************************
--*                                                            *
--/ 10.18   Physical Cues Output Interface Types               *
--*                                                            *
--**************************************************************

--****************************************
--*
--/ 10.18.1   Aircraft/Simulator Specific Physical Cues Types
--*
--****************************************
```

```
-- Names of systems which may be driven through motion
type Motion_Related_System is
     (G_Seat);


--*****************************************
--*
--/ 10.18.2   Aircraft/Simulator Reusable Physical Cues Types
--*
--*****************************************

type Operational_Status is (
   Ready,
   Not_Ready);

type Safety_Status is (
   All_Safety_Interlocks_Closed,
   Safety_Interlocks_Open);


--*****************************************
--*
--/ 10.18.3   Physical Cues Segment Output Records
--*
--*****************************************


--***************Function:
--*
--/ 10.18.3.1   Environmental Sound
--*

-- NONE


--***************Function:
--*
--/ 10.18.3.2   Anti 'G' Suit
--*
```

```
-- NONE


--****************Function:
--*
--/ 10.18.3.3   'G' Seat
--*

-- NONE


--****************Function:
--*
--/ 10.18.3.4   Motion Geometry
--*

-- NONE


--****************Function:
--*
--/ 10.18.3.5   Motion Cue
--*

-- NONE


--****************Function:
--*
--/ 10.18.3.6   Motion Base
--*

-- NONE


--****************Function:
--*
--/ 10.18.3.7   Vibration and Buffet
--*

-- NONE
```

```
--***************Function:
--*
--/ 10.18.3.8   Physical Cue Support
--*

-- SEND-ON-CHANGE

type Motion_Related_System_Status is
 record
   System_Name               : Motion_Related_System;
   System_Operational_State  : Operational_Status;
   System_Safety_State       : Safety_Status;
end record;

-- See Control_Types for responses to IOS

--*****************************************
--*
--/ 10.18.4   Physical Cues Representation Specs
--*
--*****************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes     : constant := 8; -- Bits per byte
   Byte_Size : constant := 1 * Bytes;
   Word_Size : constant := 4 * Bytes;

   for Motion_Related_System'size use Byte_Size;
   for Operational_Status'size use Byte_Size;
   for Safety_Status'size use Byte_Size;

   for Motion_Related_System_Status use
    record
      System_Name              at 0 range 0..Byte_Size-1;
      System_Operational_State at 1 range 0..Byte_Size-1;
      System_Safety_State      at 2 range 0..Byte_Size-1;
```

```
    end record;
    for Motion_Related_System_Status'size use 3 * Byte_Size;

--*
end Physical_Cues_Output_Interface_Types;
```

```
---------------------------------------------------------------------
-- %Z% Unit Name:       Physical_Cues_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:       Package Spec (no body)
-- %Z% Unit ID:         (tbd)
-- %Z% Author:          Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
---------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Physical_Cues segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
-- of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
with Physical_Cues_Output_Interface_Types;
with Control_Types;
```

```
--
package Physical_Cues_Output_Interface is
--
--*************************************************************
--*                                                           *
--/ 10.19   Physical Cues Output Interface                    *
--*                                                           *
--*************************************************************


--***************Function:
--*
--/ 10.19.1   Environmental Sound
--*

-- NONE


--***************Function:
--*
--/ 10.19.2   Anti 'G' Suit
--*

-- NONE


--***************Function:
--*
--/ 10.19.3   'G' Seat
--*

-- NONE


--***************Function:
--*
--/ 10.19.4   Motion Geometry
--*

-- NONE
```

```
--***************Function:
--*
--/ 10.19.5   Motion Cue
--*

-- NONE


--***************Function:
--*
--/ 10.19.6   Motion Base
--*

-- NONE


--***************Function:
--*
--/ 10.19.7   Vibration and Buffet
--*

-- NONE


--***************Function:
--*
--/ 10.19.8   Physical Cue Support
--*

-- SEND-ON-CHANGE OUTPUTS


Motion_Related_System_State :
   Physical_Cues_Output_Interface_Types.
   Motion_Related_System_Status;
--
-- Destination : IOS
--

Physical_Cues_Segment_Simulation_State_Response :
```

```
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Physical_Cues_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Physical_Cues_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Physical_Cues_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Physical_Cues_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Physical_Cues_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--
```

```
Physical_Cues_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--

--*
end Physical_Cues_Output_Interface;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Visual_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Visual segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
--
package Visual_Output_Interface_Types is
--
--*************************************************************
--*                                                           *
--/ 10.20   Visual System Interface Types                     *
--*                                                           *
--*************************************************************

--**************************************
```

```
--*
--/ 10.20.1   Aircraft/Simulator Specific Visual Types
--*
--******************************************

-- Declare all D/Os in the flight station that this segment will
-- turn on or off
type Visual_Discrete_Outputs is
   (To_Be_Determined);


--******************************************
--*
--/ 10.20.2   Aircraft/Simulator Reusable Visual Types
--*
--******************************************

Number_Of_Visual_Discrete_Outputs : constant :=
   Visual_Discrete_Outputs'pos (Visual_Discrete_Outputs'last) -
   Visual_Discrete_Outputs'pos (Visual_Discrete_Outputs'first) + 1;

type Visual_Hydraulic_Component_Flow_Array is
   array (Global_Message_Types.Visual_System_Hydraulic_Component) of
   Engineering_Units.Gal_Per_Min;


--******************************************
--*
--/ 10.20.3   Visual Segment Output Records
--*
--******************************************


--***************Function:
--*
--/ 10.20.3.1   Image Generation
--*

--NONE
```

```
--***************Function:
--*
--/ 10.20.3.2   Moving model
--*

--NONE


--***************Function:
--*
--/ 10.20.3.3   Visual Scene Environment
--*

--NONE


--***************Function:
--*
--/ 10.20.3.4   Lighting
--*

--NONE


--***************Function:
--*
--/ 10.20.3.5   Mission Computer/Display Processor Interface
--*

--NONE


--***************Function:
--*
--/ 10.20.3.6   Visual Crew Station Interface
--*

--SEND-ON-CHANGE

--
-- Depending on the implementation, this segment will either send a
```

```
-- number of these messages:
--
type Visual_Discrete_Output_And_State is record
   ID    : Visual_Discrete_Outputs;
   State : Base_Types.Discrete_State;
 end record;

--
-- ...or it will collect the discretes into an array;
--
subtype Visual_Discrete_Output_Count is
   Base_Types.Unsigned_Integer_32
   range 1..Number_Of_Visual_Discrete_Outputs;

type Visual_Discrete_Output_Array is array (
   Visual_Discrete_Output_Count) of
   Visual_Discrete_Output_And_State;

--
-- ...and send the ones which have changed in one of these messages:
--
type Visual_Discrete_Output_List is
 record
   Number_Of_DOs    : Visual_Discrete_Output_Count;
   Discrete_Outputs : Visual_Discrete_Output_Array;
 end record;



--***************Function:
--*
--/ 10.20.3.7   Visual Aircraft Systems Interface
--*

type Visual_Aircraft_Systems_Interface_Eighth_Rate is
 record
   Electrical_Loads          : Global_Message_Types.
                               Aircraft_Electrical_Bus_Load_Array;
                               --FS
   Hydraulic_Component_Flows : Visual_Hydraulic_Component_Flow_Array;
                               --FS
```

```
end record;


--***************Function:
--*
--/ 10.20.3.8    Visual Display Systems
--*

--NONE


--***************Function:
--*
--/ 10.20.3.9   Radar Database
--*

-- See Service_Function_Types


--***************Function:
--*
--/ 10.20.3.10  Visual Database
--*

-- See Service_Function_Types


--***************Function:
--*
--/ 10.20.3.11  Spatial Relations
--*

-- See Service_Function_Types


--***************Function:
--*
--/ 10.20.3.12  Occulting
--*

-- See Service_Function_Types
```

```
--***************Function:
--*
--/ 10.20.3.13  Visual Support
--*

--SEND-ON-CHANGE

-- See Control_Types for responses to IOS


--*****************************************
--*
--/ 10.20.4   Visual Representation Specs
--*
--*****************************************
private

   --
   -- Declarations to make representation specs more readable
   --
   Bytes     : constant := 8; -- Bits per byte
   Byte_Size : constant := 1 * Bytes;
   Word_Size : constant := 4 * Bytes;

   -- 10.20.3.7
   Visual_Discrete_Output_And_State_Size : constant := 2 * Bytes;
   for Visual_Discrete_Output_And_State'size use
      Visual_Discrete_Output_And_State_Size;

   Visual_Discrete_Output_Array_Size : constant :=
      Visual_Discrete_Output_And_State_Size *
      Number_Of_Visual_Discrete_Outputs;
   for Visual_Discrete_Output_Array'size use
      Visual_Discrete_Output_Array_Size;

   for Visual_Discrete_Output_List use
    record
      Number_Of_DOs    at 0
                       range 0..Word_Size-1;
```

```
      Discrete_Outputs at Word_Size/Bytes
                        range 0..
                        Visual_Discrete_Output_Array_Size-1;
   end record;
  for Visual_Discrete_Output_List'size use
     Word_Size + Visual_Discrete_Output_Array_Size;

  -- 10.20.3.8
  Number_Of_Visual_System_Hydraulic_Components : constant :=
     Global_Message_Types.Visual_System_Hydraulic_Component'pos (
     Global_Message_Types.Visual_System_Hydraulic_Component'last) -
     Global_Message_Types.Visual_System_Hydraulic_Component'pos (
    Global_Message_Types.Visual_System_Hydraulic_Component'first) + 1;

  Visual_Hydraulic_Component_Flow_Array_Size : constant :=
     Number_Of_Visual_System_Hydraulic_Components * Word_Size;
  for Visual_Hydraulic_Component_Flow_Array'size use
     Visual_Hydraulic_Component_Flow_Array_Size;

  for Visual_Aircraft_Systems_Interface_Eighth_Rate use
   record
     Electrical_Loads
        at 0
        range 0..Global_Message_Types.
              Aircraft_Electrical_Bus_Load_Array_Size-1;
     Hydraulic_Component_Flows
        at Global_Message_Types.
           Aircraft_Electrical_Bus_Load_Array_Size/Bytes
        range 0..Visual_Hydraulic_Component_Flow_Array_Size-1;
   end record;
  for Visual_Aircraft_Systems_Interface_Eighth_Rate'size use
     Global_Message_Types.Aircraft_Electrical_Bus_Load_Array_Size +
     Visual_Hydraulic_Component_Flow_Array_Size;

--*
end Visual_Output_Interface_Types;
```

```
---------------------------------------------------------------------------
-- %Z% Unit Name:        Visual_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
---------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by the
--  Visual segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the functions
--  in this segment will not be performed, based on simulator
requirements.
--  The messages associated with these functions need not be sent, and
--  should therefore be deleted or commented out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which receive
--  this message.  These comments should be modified to account for
--  (a) the presence or absence of other segments, and (b) the
requirements
--  of the other segments for data.  For example, if segment X is absent,
--  then the notation that segment X is a destination of a given
--  message should be removed.  Similarly, when segment Y does not
require
--  the data in a given message, then the notation that segment Y is a
--  destination for that message should be removed.
--
--  When the segment abbreviations have all been removed for a message,
--  it is clear that this message need not be sent, and the message
--  object declaration itself may be commented out or deleted.
--
--  The four service functions: Radar Database, Visual Database, Spatial
--  Relations and Occulting have messages which must each be sent by one
```

```
--  and only one segment.  Only one of the following three segments:
--  Environment, Radar or Visual, may send these messages.  Modify the
--  output interface packages for each of these three segments in
--  accordance with the assignment of the functions to the segments,
--  commenting or uncommenting declarations accordingly.
--
with Visual_Output_Interface_Types;
with Control_Types;
with Service_Function_Types;
--
package Visual_Output_Interface is
--
--***************************************************************
--*                                                              *
--/ 10.21   Visual System Output Interface                       *
--*                                                              *
--***************************************************************


--***************Function:
--*
--/ 10.21.1   Image Generation
--*

--NONE


--***************Function:
--*
--/ 10.21.2   Moving Model
--*

--NONE


--***************Function:
--*
--/ 10.21.3   Visual Scene Environment
--*

--NONE
```

```
--***************Function:
--*
--/ 10.21.4   Lighting
--*

--NONE


--***************Function:
--*
--/ 10.21.5   Mission Computer/Display Processor Interface
--*

--NONE


--***************Function:
--*
--/ 10.21.6    Visual Crew Station Interfacing
--*

--SEND-ON-CHANGE OUTPUTS

Visual_Discrete_Output_Change :
   Visual_Output_Interface_Types.
   Visual_Discrete_Output_List;
--
-- Destination: FS
--


--***************Function:
--*
--/ 10.21.7    Visual Aircraft Systems Interface
--*

Visual_Aircraft_Systems_Interface_Eighth_Rate_Output :
   Visual_Output_Interface_Types.
   Visual_Aircraft_Systems_Interface_Eighth_Rate;
```

```
--
-- Destination: FS
--


--***************Function:
--*
--/ 10.21.8   Visual Display Systems
--*

--NONE


--***************Function:
--*
--/ 10.21.9  Radar Database
--*

--NONE


--***************Function:
--*
--/ 10.21.10  Visual Database
--*

--NONE


--***************Function:
--*
--/ 10.21.11  Spatial Relations
--*

--SEND-ON-CHANGE OUTPUTS

-- Position_Range_Change :
--    Service_Function_Types.
--    Position_Range_Update;
-- Destination: RDR, NAV, ENV
--
```

```
-- Assigned to RDR

-- Groundspeed_Change :
--    Service_Function_Types.
--    Groundspeed_Update;
-- Destination: RDR, NAV, ENV
--
-- Assigned to RDR

-- Ownship_Height_Above_Terrain_Max_Rate_Outputs :
--    Service_Function_Types.
--    Ownship_Height_Above_Terrain;
-- Destination: ENV, NAV, WPN, PHC, RDR, FD
--
-- Assigned to ENV

-- Moving_Models_Height_Above_Terrain_Max_Rate_Outputs :
--    Service_Function_Types.
--    Moving_Models_Height_Above_Terrain;
-- Destination: ENV, NAV, WPN, PHC, RDR, FD
--
-- Assigned to ENV


--***************Function:
--*
--/ 10.21.12  Occulting
--*

--SEND-ON-CHANGE OUTPUTS

Occulting_Status_Change :
   Service_Function_Types.
   Occulting_Status_Update;
--
-- Destination: ENV, NAV, EW, RDR
--


--***************Function:
--*
```

```
--/ 10.21.13  Visual Support
--*

-- SEND-ON-CHANGE OUTPUTS

Visual_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Visual_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Visual_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Visual_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Visual_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Visual_On_Line_Diagnostic_Response :
```

```
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Visual_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--

--*
end Visual_Output_Interface;
```

```
------------------------------------------------------------------------
-- %Z% Unit Name:        IOS_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   3 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the IOS segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
--  All responses to IOS messages have been collected together in
--  Control_Types.  Where a type is used in both an IOS command and
--  the response to the command, that type is defined in Control_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
with Control_Types;
with Moving_Model_Types;
--
package IOS_Output_Interface_Types is
--
--*************************************************************
```

```
--*                                                                    *
--/ 10.22   IOS Output Interface Types                                 *
--*                                                                    *
--**********************************************************************

Bytes : constant := 8;

--*****************************************
--*
--/ 10.22.1   Aircraft/Simulator Specific IOS Types
--*
--*****************************************

Maximum_Snapshot_Number   : constant := 10;
Maximum_Record_Number     : constant := 10;
Maximum_Recording_Session : constant := 10.0; -- Minutes

type Simulation_States is (
   Mission_Generation,
   Training,
   Shutdown,
   Remote_Controlled_Diagnostic,
   Reset,
   Memory_Erase);
for Simulation_States'size use 16;

type Training_Modes is (
   Initialize,
   Run,
   Align_To_Approach,
   Align_To_Departure,
   Total_Freeze,
   End_Training);
for Training_Modes'size use 16;

type Quick_Action_Task is (
   Engine_Quick_Start,       --PRO
   INS_Rapid_Alignment,      --NAV
   Perfect_INU);             --NAV

type Run_Mode_Freeze is (
```

```
   Flight_Freeze,  --FD
   Position_Freeze,  --FD, NAV
   Altitude_Freeze,  --FD
   Pitch_Freeze,  --FD
   Yaw_Freeze,  --FD
   Airspeed_Freeze,  --FD
   Weapon_Freeze,  --WPN
   Heading_Freeze,  --FD
   Fuel_Quantity_Freeze);  --FS

subtype Visual_Eyepoint is Global_Message_Types.Crew_Station
   range Global_Message_Types.Pilot..Global_Message_Types.Pilot;

type IOS_Discrete_Outputs is (
   Control_Loading,     --FC
   G_Seat,              --PHC
   Visual,              --VIS
   Crash_Override);     --FD

type IOS_Discrete_Commands is (
   Aircraft_On_Jacks,              --FD, FC
   Instructor_Press_To_Talk_Flag,  --NAV
   Tanker_Position_Freeze);        --FD

type Playback_Speeds is (
   Double_Speed,
   Full_Speed,
   Half_Speed,
   Quarter_Speed);

type Time_Parameter is (
   Time_Of_Day,           --NAV, VIS
   Mission_Clock,         --NAV,VIS,EW,WPN,PRO,FC,FD,FS,RDR
   Mission_Elapsed_Time,  --NAV,VIS,EW,WPN,PRO,FC,FD,FS,RDR
   Greenwich_Mean_Time);  --EW

type Visual_Model_Database is
     (Default_Model);  --VIS

type Adjustable_Lighting is (
   Approach_Lights,                  --VIS
```

```
   Centerline_Lights,                      --VIS
   Directional_Lights,                     --VIS
   ICCS_Lights,                            --VIS
   Ramp_Lights,                            --VIS
   Runway_End_Identifier_Lights,           --VIS
   Runway_Edge_Lights,                     --VIS
   Strobe_Lights,                          --VIS
   Taxiway_Lights,                         --VIS
   Touchdown_Zone,                         --VIS
   VASI_Lights,                            --VIS
   Ambient_Illumination,                   --VIS, NAV
   Horizon_Brightness);                    --VIS

type Environmental_Set is (
   Cloudy,                      --VIS
   Lightning,                   --VIS
   Patchy_Fog,                  --VIS
   Rain,                        --VIS, PHC
   Scud,                        --VIS
   Snow_On_Runway,              --VIS
   Ice_On_Runway,               --VIS
   Runway_Roughness);           --PHC, FD

type Visual_Range is (
   Runway_Visual_Range_RVR,     --VIS
   Flight_Visual_Range_FVR,     --VIS
   General_Visibility);        --VIS

type Cloud_Adjustment is (
   Top,          --VIS
   Bottom);      --VIS

type Icing_Surface is (
   Fuselage,         --FD
   Engine,           --FD, PRO
   Left_Wing,        --FD
   Right_Wing,       --FD
   Tail);            --FD

type Resettable_Value is (
   Fire_Agent_Bottle,               --FS
```

```
      Battery_Charge,                  --FS
      Oxygen_Bottle,                   --FS
      Emergency_Oxygen_Bottle,         --FS
      Cabin_Temperature,               --FS
      Cabin_Altitude,                  --FS
      Hydraulic_Fluid_Temperature,     --FS
      Fuel_Temperature,                --FS
      Engine_Oil_Temperature,          --PRO
      Exhaust_Gas_Temperature,         --PRO
      Brake_Temperature,               --FC
      Engine_Oil_Quantity,             --PRO
      Hydraulic_Fluid_Quantity,        --FS
      Auxiliary_Power_Unit,            --PRO
      Ownship_Weapon_Stores);          --WPN, FS

   type Fuel_Quantity is (
      Left_Wing,                  --FS
      Right_Wing,                 --FS
      F_1,                        --FS
      F_2,                        --FS
      Aft_Reservoir,              --FS
      Fwd_Reservoir,              --FS
      A_1,                        --FS
      Left_External,              --FS
      Right_External,             --FS
      Centerline_External,        --FS
      Total_Fuel_Quantity);       --FS

   type External_Connection is (
      External_AC,          --FS, PHC
      External_DC,          --FS, PHC
      Ground_Cart_Air,      --FS, PHC
      Boom);                --FS, PHC, FD

   type Connection_State is (Connected, Disconnected);

   type Temperature_Profile is (
      Standard_Day,    --FD
      Hot,             --FD
      Cold,            --FD
      Polar,           --FD
```

```
   Tropical);     --FD

type Runway_Surface_Condition is digits 1 range 0.0..3.0;

subtype Runway_Condition_Reading is Base_Types.Unsigned_Integer_32
   range 0..23;
-- these take the place of snow, ice, sand etc.

subtype Temperature_Lapse_Rate is
   Engineering_Units.Degrees_C range -5.0..1.0;  --per 1000 ft.

subtype Wind_Speed_Lapse_Rate is
   Engineering_Units.Knots range -10.0..10.0;  -- per 1000 ft.

subtype Wind_Direction_Lapse_Rate is
   Engineering_Units.Degrees range 0.0..5.0;  -- per second

type Temperature_Set is (
   Sea_Level,
   Ground_Level,
   Outside_Air);

type Pressure_Set is (
   Sea_Level,
   Ground_Level,
   Outside_Air);

type Turbulence is (
   Cobblestone,
   Chop,
   Jet_Upset,
   Rough_Air);

type Wind_Set is (
   Surface,
   Steady_State);

type Wind_Profile is
   (To_Be_Determined);

type Instructor_Controllable_Parameter is (
```

```
   Mach_Number,                    --FD
   Net_Thrust,                     --PRO
   Rate_Of_Climb,                  --FD
   A_R_Fuel_Flow,                  --FS
   Center_Of_Gravity,              --FD
   Gross_Weight,                   --FD
   Airspeed);                      --FD

type Slewable_Parameter is (
   Altitude,
   Airspeed,
   Heading,
   Latitude,
   Longitude);

subtype Moving_Model_Complexity is Base_Types.Signed_Integer_32 range
0..10;


--******************************************
--*
--/ 10.22.2   Aircraft/Simulator Reusable IOS Types
--*
--******************************************

subtype Clock_Ticks is Base_Types.Signed_Integer_32
   range 1..16#7fffffff#;

type Segment_Selection_Array is
   array (Control_Types.Segment_Names) of Base_Types.Discrete_State;

type Freeze_State is (
   Frozen,
   Un_Frozen);

subtype Recording_Number is Base_Types.Unsigned_Integer_16
   range 1..Maximum_Record_Number;

subtype Recording_Session_Time is Engineering_Units.Minutes
   range 0.0..Maximum_Recording_Session;

subtype Snapshot_IDs is Base_Types.Unsigned_Integer_16 range
```

```
   1..Maximum_Snapshot_Number;

type Task_Commands is (
   Initialize_Task,
   Execute_Task,
   Hold_Task,
   Resume_Task,
   Abort_Task);

type Snapshot_Commands is (Snapshot, Recall);

type On_Line_Diagnostic_Array is
  array (Control_Types.On_Line_Diagnostics) of Base_Types.Discrete_State;

type Time is
 record
   Hours   : Base_Types.Unsigned_Integer_32 range 0..24;
   Minutes : Base_Types.Unsigned_Integer_32 range 0..60;
   Seconds : Engineering_Units.Seconds;
end record;

type Month_In_A_Year is (
   Jan, Feb, Mar, Apr, May, Jun,
   Jul, Aug, Sep, Oct, Nov, Dec);

type Platform_Radio is (
   UHF,
   VHF,
   TACAN,
   HF);


--*****************************************
--*
--/ 10.22.3   Instructor/Operator Station Segment Output Records
--*
--*****************************************


--***************Function:
--*
```

```
--/ 10.22.3.1   Simulator Control
--*


--***************Component:
--*
--/ 10.22.3.1.1   Simulation State
--*

type Segment_Simulation_State_Command is
 record
   Segments_Affected : Segment_Selection_Array;
   Control_Command   : Simulation_States;
end record;


--***************Component:
--*
--/ 10.22.3.1.2   Training Mode
--*
type Segment_Training_Mode_Command is
 record
   Segments_Affected : Segment_Selection_Array;
   Control_Command   : Training_Modes;
end record;


--***************Component:
--*
--/ 10.22.3.1.3   Quick Action Task
--*
type Quick_Action_Task_Command is
 record
   ID : Quick_Action_Task;
end record;


--***************Component:
--*
--/ 10.22.3.1.4   Run Mode Freeze
--*
```

```
type Run_Mode_Freeze_Command is
 record
   ID    : Run_Mode_Freeze;
   State : Freeze_State;
end record;


--***************Component:
--*
--/ 10.22.3.1.5   Visual Eyepoint Active
--*
-- See Aircraft/Simulator Specific Types above


--***************Component:
--*
--/ 10.22.3.1.6   Simulator Control Discrete
--*
type IOS_Discrete_Output_And_State is
 record
   ID    : IOS_Discrete_Outputs;
   State : Base_Types.Discrete_State;
end record;


--***************Component:
--*
--/ 10.22.3.1.7   Performance Test
--*
type Performance_Test_Command is
 record
   Segments_Affected : Segment_Selection_Array;
   Performance_Test  : Control_Types.Performance_Tests;
   Task_Command      : Task_Commands;
end record;


--***************Component:
--*
--/ 10.22.3.1.8   Snapshot / Replay
--*
```

```
type Snapshot_Command is
 record
   Command      : Snapshot_Commands;
   Snapshot_ID : Snapshot_IDs;
end record;


--***************Component:
--*
--/ 10.22.3.1.9   Record / Playback
--*
type Record_Command is
 record
   Record_Playback_ID : Recording_Number;
   Session_Time       : Recording_Session_Time;
end record;

type Playback_Command is
 record
   Record_Playback_ID  : Recording_Number;
   Playback_Start_Time : Recording_Session_Time := 0.0;
   Speed               : Playback_Speeds := Full_Speed;
end record;


--***************Component:
--*
--/ 10.22.3.1.10  Off Line Diagnostic
--*
type Off_Line_Diagnostic_Command is
 record
   Segments_Affected   : Segment_Selection_Array;
   Off_Line_Diagnostic : Control_Types.Off_Line_Diagnostics;
   Task_Command        : Task_Commands;
end record;


--***************Component:
--*
--/ 10.22.3.1.11  Remote Controlled Diagnostic
--*
```

```
type Remote_Controlled_Diagnostics_Command is
 record
   Segments_Affected             : Segment_Selection_Array;
  Remote_Controlled_Diagnostic : Control_Types.Remote_Controlled_Diagnostics;
   Task_Command                  : Task_Commands;
end record;


--***************Component:
--*
--/ 10.22.3.1.12  On Line Diagnostic
--*
type On_Line_Diagnostic_Command is
 record
   Segments_Affected              : Segment_Selection_Array;
   Response_Rate                  : Engineering_Units.Minutes := 0.0;
   On_Line_Diagnostics_Requested : On_Line_Diagnostic_Array;
end record;


type Timing_Commands is
record
   Segment   : Control_Types.Segment_Names;
   Subsystem : Control_Types.Subsystems;
   Component : Control_Types.Components;
   Command   : Control_Types.Structural_Element_To_Time;
end record;


--***************Component:
--*
--/ 10.22.3.1.13  Time Change
--*
type Time_Request is
 record
   Time_Name : Time_Parameter;
   New_Time  : Time;
end record;


--***************Component:
```

```
--*
--/ 10.22.3.1.14  Multiple Simulator Environment
--*
type Multiple_Simulator_Environment_Command is (
   Disconnect,
   Connect);



--***************Function:
--*
--/ 10.22.3.2   Ownship Status and Control
--*

--***************Component:
--*
--/ 10.22.3.2.1   Clock Tick (Current Frame)
--*
type Clock_Tick_Messages is
record
   Clock_Tick               : Clock_Ticks;
   Current_Simulation_Frame : Control_Types.Simulation_Frames;
end record;

--SEND-ON-CHANGE

--***************Component:
--*
--/ 10.22.3.2.2   Parameter_Slew
--*
type Parameter_Slew_Demand is
 record
   Parameter : Slewable_Parameter;
   Slew_Rate : Base_Types.Float_32;  -- this is a multiplier
end record;


--***************Component:
--*
--/ 10.22.3.2.3   Add Cargo
--*
```

```
type Add_Cargo_Command is
 record
   Cargo_Weight   : Engineering_Units.Pounds;
   Cargo_Location : Engineering_Units.Linear_Position_Components;
end record;


--***************Component:
--*
--/ 10.22.3.2.4   External Connection
--*
type External_Connection_Command is
 record
   Connection          : External_Connection;
   Demanded_Connection : Connection_State;
end record;


--***************Component:
--*
--/ 10.22.3.2.5   Parameter Change
--*
type Parameter_Change_Request is
 record
   Control_Parameter : Instructor_Controllable_Parameter;
   Value_Requested   : Base_Types.Float_32;
end record;


--***************Component:
--*
--/ 10.22.3.2.6   Hydraulic Fluid Quantity Change
--*
type Hydraulic_Fluid_Quantity_Change is
 record
   ID                : Global_Message_Types.Aircraft_Hydraulic_Reservoir;
   Demanded_Quantity : Engineering_Units.Gallons;
end record;
```

```
--***************Component:
--*
--/ 10.22.3.2.7   Fuel Quantity Change
--*
type Fuel_Quantity_Change is
 record
   ID                 : Fuel_Quantity;
   Demanded_Quantity : Engineering_Units.Pounds;
end record;


--***************Component:
--*
--/ 10.22.3.2.8   Reset
--*
type Reset_Demand is
 record
   ID : Resettable_Value;
end record;


--***************Component:
--*
--/ 10.22.3.2.9   Icing Level
--*
type Icing_Level_Adjustment is
 record
   ID              : Icing_Surface;
   Demanded_Value : Engineering_Units.Zero_To_Ten;
end record;


--***************Component:
--*
--/ 10.22.3.2.10  Engine Oil Quantity
--*
type Engine_Oil_Quantity_Adjustment is
 record
   ID              : Global_Message_Types.Aircraft_Engine;
   Demanded_Value : Engineering_Units.Gallons;
end record;
```

```
--***************Component:
--*
--/ 10.22.3.2.11  Discrete Command
--*
type IOS_Discrete_Command_And_State is
 record
   Command : IOS_Discrete_Commands;
   State   : Base_Types.Discrete_State;
end record;


--***************Function:
--*
--/ 10.22.3.3   Ownship Malfunction
--*

-- See Control_Types for definition of Malfunction_Demand


--***************Function:
--*
--/ 10.22.3.4   Ownship Controls Disagreement
--*

--NONE


--***************Function:
--*
--/ 10.22.3.5   Navigation/Communication Status and Control
--*

--SEND-ON-CHANGE

-- See Global_message_Types for definition of Waypoint_Change and
-- Earth_Position_Components

type Platform_Radio_Set is
 record
```

```
   ID        : Moving_Model_Types.Moving_Model_ID;
   Frequency : Engineering_Units.Hertz;
   Radio     : Platform_Radio;
   Valid     : Base_Types.Sim_Boolean;
end record;

type Training_Area_Boundaries is
 record
   N_W_Corner : Engineering_Units.Lat_Long_Location;
   N_E_Corner : Engineering_Units.Lat_Long_Location;
   S_W_Corner : Engineering_Units.Lat_Long_Location;
   S_E_Corner : Engineering_Units.Lat_Long_Location;
end record;


--****************Function:
--*
--/ 10.22.3.6   Physical/Natural Environment Status and Control
--*

--SEND-ON-CHANGE

type Wind_Shear is
 record
   Top_Direction     : Engineering_Units.Degrees;
   Top_Velocity      : Engineering_Units.Knots;
   Bottom_Direction  : Engineering_Units.Degrees;
   Bottom_Velocity   : Engineering_Units.Knots;
   Altitude          : Engineering_Units.Feet;
end record;

type Wind_Direction_Set is
 record
   ID        : Wind_Set;
   Direction : Engineering_Units.Degrees;
end record;

type Wind_Speed_Set is
 record
   ID    : Wind_Set;
   Speed : Engineering_Units.Knots;
```

```
end record;

type Wind_Position is
 record
   ID       : Wind_Profile;
   Position : Engineering_Units.Earth_Position_Components;
end record;

type Wind_Intensity is
 record
   ID        : Wind_Profile;
   Intensity : Engineering_Units.Zero_To_Ten;
end record;

type Turbulence_Demand is
 record
   Selected_Type : Turbulence;
   Magnitude     : Engineering_Units.Zero_To_Ten;
end record;

-- See 10.22.1 above for definitions of Runway_Surface_Condition
-- and Runway_Condition_Reading

type Pressure_Demand is
 record
   ID        : Pressure_Set;
   Set_Value : Engineering_Units.Inches_Hg;
end record;

type Temperature_Demand is
 record
   ID        : Temperature_Set;
   Set_Value : Engineering_Units.Degrees_C;
end record;

-- See 10.22.1 above for definition of Visual_Model_Database

-- See Moving_Model_Types for definition of Moving_Model_Dynamic_Data

type Thunderstorm_Intensity_Set is
 record
```

```
   ID              : Moving_Model_Types.Moving_Model_ID;
   Storm_Intensity : Engineering_Units.Zero_To_Ten;
end record;

type Lighting_Adjustment is
 record
   Lighting_Element : Adjustable_Lighting;
   Intensity        : Engineering_Units.Zero_To_Ten;
end record;

type Environment_Adjustment is
 record
   ID        : Environmental_Set;
   Intensity : Engineering_Units.Zero_To_Ten;
end record;

type Visual_Range_Adjustment is
 record
   ID        : Visual_Range;
   Range_Set : Engineering_Units.Nautical_Miles;
end record;

type Cloud_Level_Adjustment is
 record
   Cloud_Level       : Cloud_Adjustment;
   Adjustment_Height : Engineering_Units.Feet;
end record;

type Year_Time is
 record
   Day   : Base_Types.Unsigned_Integer_32 range 1..31;
   Month : Month_In_A_Year;
   Year  : Base_Types.Unsigned_Integer_32 range 1950..2000;
end record;

-- See 10.22.1 above for definitions of Temperature_Profile,
-- Temperature_Lapse_Rate, Wind_Speed_Lapse_Rate, and Wind_
-- Direction_Lapse_Rate


--***************Function:
```

```
--*
--/ 10.22.3.7   Tactical and Mission Environment Status and Control
--*

--SEND-ON-CHANGE

-- See Moving_Model_Types for definitions of:
--   Moving_Model_Dynamic_Data
--   Moving_Model_Deactivation
--   Chaff_Moving_Model_Unique_Data
--   Flare_Moving_Model_Unique_Data
--   Decoy_Moving_Model_Unique_Data
--   Platform_Moving_Model_Unique_Data
--   Tanker_Moving_Model_Unique_Data
--   Moving_Model_IFF_Data

type Emitter_Set is
 record
   ID           : Moving_Model_Types.Moving_Model_ID;
   Emitter_Data : Moving_Model_Types.Emitter_Frequency_Data;
   State        : Base_Types.Discrete_State;
end record;

type Battle_Damage_Set is
 record
   ID             : Moving_Model_Types.Moving_Model_ID;
   Desired_Damage : Moving_Model_Types.Model_Damage_Data;
end record;

type Model_Lighting_Set is
 record
   ID               : Moving_Model_Types.Moving_Model_ID;
   Desired_Lighting : Moving_Model_Types.Model_Lighting;
   State            : Base_Types.Discrete_State;
end record;

type Articulated_Device_Set is
 record
   ID            : Moving_Model_Types.Moving_Model_ID;
   Device_Change : Moving_Model_Types.Articulated_Part_Data;
end record;
```

```
type Weapon_Load_Set is
 record
   ID           : Moving_Model_Types.Moving_Model_ID;
   Desired_Load : Global_Message_Types.Weapon_Station_Loading;
end record;

type Weapon_Fire_Command is
 record
   ID           : Moving_Model_Types.Moving_Model_ID;
   Fired_Weapon : Global_Message_Types.Station_Weapon_Load;
end record;

type Moving_Model_Complexity_Set is
 record
   ID         : Moving_Model_Types.Moving_Model_ID;
   Complexity : Engineering_Units.Zero_To_Ten;
end record;

--***************Function:
--*
--/ 10.22.3.8   Crew Station Performance Monitoring and Measurement
--*

--NONE


--***************Function:
--*
--/ 10.22.3.9   Instructor/Operator Station Segment Support
--*

--NONE

--*****************************************
--*
--/ 10.22.4   IOS Representation Specs
--*
--*****************************************
private
```

```
--
-- Declarations to make representation specs more readable
--
Byte_Size     : constant := 1 * Bytes;
Halfword_Size : constant := 2 * Bytes;
Word_Size     : constant := 4 * Bytes;

-- 10.22.3.1.1
Discrete_State_Size : constant := 8;
Performance_Tests_Size : constant :=
   Control_Types.Performance_Tests_Size;

for Segment_Simulation_State_Command use
 record
   Segments_Affected at  0 range 0..(12 * Bytes)-1;
   Control_Command   at 12 range 0..(2 * Bytes)-1;
 end record;
for Segment_Simulation_State_Command'size use 14 * Bytes;

-- 10.22.3.1.2
for Segment_Training_Mode_Command use
 record
   Segments_Affected at  0 range 0..(12 * Bytes)-1;
   Control_Command  at 12 range 0..(2 * Bytes)-1;
end record;
for Segment_Training_Mode_Command'size use 14 * Bytes;

-- 10.22.3.1.3
Quick_Action_Task_Size : constant := 8;
for Quick_Action_Task'size use Quick_Action_Task_Size;

for Quick_Action_Task_Command use
 record
   ID at 0 range 0..Quick_Action_Task_Size-1;
end record;
for Quick_Action_Task_Command'size use Quick_Action_Task_Size;

-- 10.22.3.1.4
Run_Mode_Freeze_Size : constant := 8;
for Run_Mode_Freeze'size use Run_Mode_Freeze_Size;
```

```
Freeze_State_Size : constant := 8;
for Freeze_State'size use Freeze_State_Size;

for Run_Mode_Freeze_Command use
 record
   ID    at 0                          range 0..Run_Mode_Freeze_Size-1;
   State at Run_Mode_Freeze_Size/Bytes range 0..Freeze_State_Size-1;
end record;
for Run_Mode_Freeze_Command'size use
   Run_Mode_Freeze_Size +
   Freeze_State_Size;

-- 10.22.3.1.6
IOS_Discrete_Outputs_Size : constant := 8;
for IOS_Discrete_Outputs'size use IOS_Discrete_Outputs_Size;

for IOS_Discrete_Output_And_State use
 record
   ID
      at 0
      range 0..IOS_Discrete_Outputs_Size-1;
   State
      at IOS_Discrete_Outputs_Size/Bytes
      range 0..Byte_Size-1;
end record;
for IOS_Discrete_Output_And_State'size use
   IOS_Discrete_Outputs_Size + Discrete_State_Size;

-- 10.22.3.1.7
Number_Of_Segments : constant :=
   Control_Types.Segment_Names'pos (
   Control_Types.Segment_Names'last) -
   Control_Types.Segment_Names'pos (
   Control_Types.Segment_Names'first) + 1;

Segment_Selection_Array_Size : constant :=
   Number_Of_Segments * Discrete_State_Size;
for Segment_Selection_Array'size use Segment_Selection_Array_Size;

Task_Commands_Size : constant := 8;
for Task_Commands'size use Task_Commands_Size;
```

```
for Performance_Test_Command use
 record
   Segments_Affected
      at 0
      range 0..Segment_Selection_Array_Size-1;
   Performance_Test
      at Segment_Selection_Array_Size/Bytes
      range 0..Performance_Tests_Size-1;
   Task_Command
      at Segment_Selection_Array_Size/Bytes +
         Performance_Tests_Size/Bytes
      range 0..Task_Commands_Size-1;
 end record;
for Performance_Test_Command'size use
   Segment_Selection_Array_Size +
   Performance_Tests_Size +
   Task_Commands_Size;

-- 10.22.3.1.8
Snapshot_Commands_Size : constant := 8;
for Snapshot_Commands'size use Snapshot_Commands_Size;
for Snapshot_Command use
 record
   Command     at 0 range 0..Snapshot_Commands_Size-1;
   Snapshot_ID at Snapshot_Commands_Size/Bytes
      range 0..Halfword_Size-1;
end record;
for Snapshot_Command'size use
   Snapshot_Commands_Size + Halfword_Size;

-- 10.22.3.1.9
for Record_Command use
 record
   Record_Playback_ID at 0 range 0..Halfword_Size-1;
   -- 2 bytes spare
   Session_Time       at Word_Size/Bytes range 0..Word_Size-1;
   end record;
for Record_Command'size use 2 * Word_Size;

Playback_Speeds_Size : constant := 8;
```

```
for Playback_Speeds'size use Playback_Speeds_Size;

for Playback_Command use
 record
   Record_Playback_ID
      at 0
      range 0..Halfword_Size-1;
   -- 2 bytes spare
   Playback_Start_Time
      at 1 * Word_Size/Bytes
      range 0..Word_Size-1;
   Speed
      at 2 * Word_Size/Bytes
      range 0..Playback_Speeds_Size-1;
end record;
for Playback_Command'size use
   2 * Word_Size + Playback_Speeds_Size;

-- 10.22.3.1.10
Off_Line_Diagnostics_Size : constant :=
   Control_Types.Off_Line_Diagnostics_Size;

for Off_Line_Diagnostic_Command use
 record
   Segments_Affected
      at 0
      range 0..Segment_Selection_Array_Size-1;
   Off_Line_Diagnostic
      at Segment_Selection_Array_Size/Bytes
      range 0..Off_Line_Diagnostics_Size-1;
   Task_Command
      at Segment_Selection_Array_Size/Bytes +
         Off_Line_Diagnostics_Size/Bytes
      range 0..Task_Commands_Size-1;
end record;
for Off_Line_Diagnostic_Command'size use
   Segment_Selection_Array_Size +
   Off_Line_Diagnostics_Size +
   Task_Commands_Size;

-- 10.22.3.1.11
```

```
Remote_Controlled_Diagnostics_Size : constant :=
   Control_Types.Remote_Controlled_Diagnostics_Size;

for Remote_Controlled_Diagnostics_Command use
 record
   Segments_Affected
      at 0
      range 0..Segment_Selection_Array_Size-1;
   Remote_Controlled_Diagnostic
      at Segment_Selection_Array_Size/Bytes
      range 0..Remote_Controlled_Diagnostics_Size-1;
   Task_Command
      at Segment_Selection_Array_Size/Bytes +
         Remote_Controlled_Diagnostics_Size/Bytes
      range 0..Task_Commands_Size-1;
end record;
for Remote_Controlled_Diagnostics_Command'size use
   Segment_Selection_Array_Size +
   Remote_Controlled_Diagnostics_Size +
   Task_Commands_Size;

-- 10.22.3.1.12
On_Line_Diagnostics_Size : constant :=
   Control_Types.On_Line_Diagnostics_Size;

Number_Of_On_Line_Diagnostics : constant :=
   Control_Types.On_Line_Diagnostics'pos (
   Control_Types.On_Line_Diagnostics'last) -
   Control_Types.On_Line_Diagnostics'pos (
   Control_Types.On_Line_Diagnostics'first) + 1;

On_Line_Diagnostic_Array_Size : constant :=
   Number_Of_On_Line_Diagnostics * Discrete_State_Size;
for On_Line_Diagnostic_Array'size use
   On_Line_Diagnostic_Array_Size;

for On_Line_Diagnostic_Command use
 record
   Segments_Affected
      at 0
      range 0..Segment_Selection_Array_Size-1;
```

```
    Response_Rate
       at Segment_Selection_Array_Size/Bytes
       range 0..Word_Size-1;
    On_Line_Diagnostics_Requested
       at Segment_Selection_Array_Size/Bytes +
          Word_Size/Bytes
       range 0..On_Line_Diagnostic_Array_Size-1;
end record;
for On_Line_Diagnostic_Command'size use
   Segment_Selection_Array_Size +
   Word_Size +
   On_Line_Diagnostic_Array_Size;

Segment_And_Tester_Names_Size : constant :=
   Control_Types.Segment_And_Tester_Names_Size;

Subsystems_Size : constant :=
   Control_Types.Subsystems_Size;

Components_Size : constant :=
   Control_Types.Components_Size;

Structural_Element_To_Time_Size : constant :=
   Control_Types.Structural_Element_To_Time_Size;

for Timing_Commands use
record
   Segment
      at 0
      range 0..Segment_And_Tester_Names_Size-1;
   Subsystem
      at Segment_And_Tester_Names_Size/Bytes
      range 0..Subsystems_Size-1;
   Component
      at Segment_And_Tester_Names_Size/Bytes +
         Subsystems_Size/Bytes
      range 0..Components_Size-1;
   Command
      at Segment_And_Tester_Names_Size/Bytes +
         Subsystems_Size/Bytes +
         Components_Size/Bytes
```

```
      range 0..Structural_Element_To_Time_Size-1;
end record;
for Timing_Commands'size use
   Segment_And_Tester_Names_Size +
   Subsystems_Size +
   Components_Size +
   Structural_Element_To_Time_Size;

-- 10.22.3.1.13
for Time use
 record
   Hours   at 0                   range 0..Word_Size-1;
   Minutes at 1 * Word_Size/Bytes range 0..Word_Size-1;
   Seconds at 2 * Word_Size/Bytes range 0..Word_Size-1;
end record;
for Time'size use 3 * Word_Size;

for Time_Parameter'size use Word_Size;

for Time_Request use
 record
   Time_Name at 0                   range 0..Word_Size-1;
   New_Time  at 1 * Word_Size/Bytes range 0..(3 * Word_Size)-1;
end record;
for Time_Request'size use 4 * Word_Size;

-- 10.22.3.1.14
for Multiple_Simulator_Environment_Command'size use 8;

-- 10.22.3.2.1
for Clock_Tick_Messages use
record
   Clock_Tick
      at 0
      range 0..Word_Size-1;
   Current_Simulation_Frame
      at Word_Size/Bytes
      range 0..Word_Size-1;
end record;
for Clock_Tick_Messages'size use 2 * Word_Size;
```

```
-- 10.22.3.2.2
Slewable_Parameter_Size : constant := Word_Size;
for Slewable_Parameter'size use Slewable_Parameter_Size;

for Parameter_Slew_Demand use
 record
   Parameter
      at 0
      range 0..Slewable_Parameter_Size-1;
   Slew_Rate
      at Slewable_Parameter_Size/Bytes
      range 0..Word_Size-1;
end record;
for Parameter_Slew_Demand'size use
   Slewable_Parameter_Size + Word_Size;

-- 10.22.3.2.3
Linear_Position_Components_Size : constant :=
   Engineering_Units.Linear_Position_Components_Size;

for Add_Cargo_Command use
 record
   Cargo_Weight
      at 0
      range 0..Word_Size-1;
   Cargo_Location
      at Word_Size/Bytes
      range 0..Linear_Position_Components_Size-1;
end record;
for Add_Cargo_Command'size use
   Word_Size + Linear_Position_Components_Size;

-- 10.22.3.2.4
External_Connection_Size : constant := 8;
for External_Connection'size use External_Connection_Size;

Connection_State_Size : constant := 8;
for Connection_State'size use Connection_State_Size;

for External_Connection_Command use
 record
```

```
      Connection
         at 0
         range 0..External_Connection_Size-1;
      Demanded_Connection
         at External_Connection_Size/Bytes
         range 0..Connection_State_Size-1;
   end record;
   for External_Connection_Command'size use
      External_Connection_Size + Connection_State_Size;

   -- 10.22.3.2.5
   Instructor_Controllable_Parameter_Size : constant := Word_Size;
   for Instructor_Controllable_Parameter'size use
      Instructor_Controllable_Parameter_Size;

   for Parameter_Change_Request use
    record
      Control_Parameter
         at 0
         range 0..Instructor_Controllable_Parameter_Size-1;
      Value_Requested
         at Instructor_Controllable_Parameter_Size/Bytes
         range 0..Word_Size-1;
   end record;
   for Parameter_Change_Request'size use
      Instructor_Controllable_Parameter_Size + Word_Size;

   -- 10.22.3.2.6
   Aircraft_Hydraulic_Reservoir_Size : constant :=
      Global_Message_Types.Aircraft_Hydraulic_Reservoir_Size;

   for Hydraulic_Fluid_Quantity_Change use
    record
      ID
         at 0
         range 0..Aircraft_Hydraulic_Reservoir_Size-1;
      Demanded_Quantity
         at Aircraft_Hydraulic_Reservoir_Size/Bytes
         range 0..Word_Size-1;
   end record;
   for Hydraulic_Fluid_Quantity_Change'size use
```

```
   Aircraft_Hydraulic_Reservoir_Size + Word_Size;

-- 10.22.3.2.7
Fuel_Quantity_Size : constant := Word_Size;
for Fuel_Quantity'size use Fuel_Quantity_Size;

for Fuel_Quantity_Change use
 record
   ID
      at 0
      range 0..Fuel_Quantity_Size-1;
   Demanded_Quantity
      at Fuel_Quantity_Size/Bytes
      range 0..Word_Size-1;
end record;
for Fuel_Quantity_Change'size use
   Fuel_Quantity_Size + Word_Size;

-- 10.22.3.2.8
Resettable_Value_Size : constant := Byte_Size;
for Resettable_Value'size use Resettable_Value_Size;

for Reset_Demand'size use Resettable_Value_Size;

-- 10.22.3.2.9
Icing_Surface_Size : constant := Word_Size;
for Icing_Surface'size use Icing_Surface_Size;

for Icing_Level_Adjustment use
 record
   ID
      at 0 range 0..Icing_Surface_Size-1;
   Demanded_Value
      at Icing_Surface_Size/Bytes
      range 0..Word_Size-1;
end record;
for Icing_Level_Adjustment'size use
   Icing_Surface_Size + Word_Size;

-- 10.22.3.2.10
Aircraft_Engine_Size : constant :=
```

```
   Global_Message_Types.Aircraft_Engine_Size;

for Engine_Oil_Quantity_Adjustment use
 record
   ID
      at 0 range 0..Aircraft_Engine_Size-1;
   Demanded_Value
      at Aircraft_Engine_Size/Bytes range 0..Word_Size-1;
end record;
for Engine_Oil_Quantity_Adjustment'size use
   Aircraft_Engine_Size + Word_Size;

-- 10.22.3.2.11
IOS_Discrete_Commands_Size : constant := Byte_Size;
for IOS_Discrete_Commands'size use IOS_Discrete_Commands_Size;

for IOS_Discrete_Command_And_State use
 record
   Command
      at 0 range 0..IOS_Discrete_Commands_Size-1;
   State
      at IOS_Discrete_Commands_Size/Bytes range 0..Byte_Size-1;
end record;
for IOS_Discrete_Command_And_State'size use
   IOS_Discrete_Commands_Size + Byte_Size;

-- 10.22.3.5
Moving_Model_ID_Size : constant :=
   Moving_Model_Types.Moving_Model_ID_Size;
Platform_Radio_Size : constant := Byte_Size;
for Platform_Radio'size use Platform_Radio_Size;

for Platform_Radio_Set use
 record
   ID
     at 0
     range 0..Moving_Model_ID_Size-1;
   Frequency
     at Moving_Model_ID_Size/Bytes
     range 0..Word_Size-1;
   Radio
```

```
      at Moving_Model_ID_Size/Bytes +
      Word_Size/Bytes
      range 0..Platform_Radio_Size-1;
    Valid
      at Moving_Model_ID_Size/Bytes +
      Word_Size/Bytes +
      Platform_Radio_Size/Bytes
      range 0..Byte_Size-1;
end record;
for Platform_Radio_Set'size use
   Moving_Model_ID_Size +
   Word_Size +
   Platform_Radio_Size +
   Byte_Size;

Lat_Long_Location_Size : constant :=
   Engineering_Units.Lat_Long_Location_Size;

for Training_Area_Boundaries use
 record
   N_W_Corner
      at 0 * Lat_Long_Location_Size/Bytes
      range 0..Lat_Long_Location_Size-1;
   N_E_Corner
      at 1 * Lat_Long_Location_Size/Bytes
      range 0..Lat_Long_Location_Size-1;
   S_W_Corner
      at 2 * Lat_Long_Location_Size/Bytes
      range 0..Lat_Long_Location_Size-1;
   S_E_Corner
      at 3 * Lat_Long_Location_Size/Bytes
      range 0..Lat_Long_Location_Size-1;
end record;
for Training_Area_Boundaries'size use
   4 * Lat_Long_Location_Size;

-- 10.22.3.6
for Wind_Shear use
 record
   Top_Direction
      at 0 * Word_Size/Bytes range 0..Word_Size-1;
```

```
   Top_Velocity
      at 1 * Word_Size/Bytes range 0..Word_Size-1;
   Bottom_Direction
      at 2 * Word_Size/Bytes range 0..Word_Size-1;
   Bottom_Velocity
      at 3 * Word_Size/Bytes range 0..Word_Size-1;
   Altitude
      at 4 * Word_Size/Bytes range 0..Word_Size-1;
end record;
for Wind_Shear'size use 5 * Word_Size;

Wind_Set_Size : constant := Word_Size;
for Wind_Set'size use Wind_Set_Size;

for Wind_Direction_Set use
 record
   ID
      at 0 range 0..Wind_Set_Size-1;
   Direction
      at Wind_Set_Size/Bytes range 0..Word_Size-1;
end record;
for Wind_Direction_Set'size use
   Wind_Set_Size + Word_Size;

for Wind_Speed_Set use
 record
   ID
      at 0 range 0..Wind_Set_Size-1;
   Speed
      at Wind_Set_Size/Bytes range 0..Word_Size-1;
end record;
for Wind_Speed_Set'size use
   Wind_Set_Size + Word_Size;

Wind_Profile_Size : constant := Word_Size;
for Wind_Profile'size use Wind_Profile_Size;

Earth_Position_Components_Size : constant :=
   Engineering_Units.Earth_Position_Components_Size;

for Wind_Position use
```

```
 record
   ID
      at 0
      range 0..Wind_Profile_Size-1;
   Position
      at Wind_Profile_Size/Bytes
      range 0..Earth_Position_Components_Size-1;
end record;
for Wind_Position'size use
   Wind_Profile_Size +
   Earth_Position_Components_Size;

for Wind_Intensity use
 record
   ID
      at 0 range 0..Wind_Profile_Size-1;
   Intensity
      at Wind_Profile_Size/Bytes range 0..Word_Size-1;
end record;
for Wind_Intensity'size use
   Wind_Profile_Size + Word_Size;

Turbulence_Size : constant := Word_Size;
for Turbulence'size use Turbulence_Size;

for Turbulence_Demand use
 record
   Selected_Type
      at 0 range 0..Turbulence_Size-1;
   Magnitude
      at Turbulence_Size/Bytes range 0..Word_Size-1;
end record;
for Turbulence_Demand'size use
   Turbulence_Size + Word_Size;

Pressure_Set_Size : constant := Word_Size;
for Pressure_Set'size use Pressure_Set_Size;

for Pressure_Demand use
 record
   ID
```

```
        at 0 range 0..Pressure_Set_Size-1;
     Set_Value
        at Pressure_Set_Size/Bytes range 0..Word_Size-1;
  end record;
  for Pressure_Demand'size use
     Pressure_Set_Size + Word_Size;

  Temperature_Set_Size : constant := Word_Size;
  for Temperature_Set'size use Temperature_Set_Size;

  for Temperature_Demand use
   record
     ID
        at 0 range 0..Temperature_Set_Size-1;
     Set_Value
        at Temperature_Set_Size/Bytes range 0..Word_Size-1;
  end record;
  for Temperature_Demand'size use
     Temperature_Set_Size + Word_Size;

  for Thunderstorm_Intensity_Set use
   record
     ID
        at 0 range 0..Moving_Model_ID_Size-1;
     Storm_Intensity
       at Moving_Model_ID_Size/Bytes range 0..Word_Size-1;
  end record;
  for Thunderstorm_Intensity_Set'size use
     Moving_Model_ID_Size + Word_Size;

  Adjustable_Lighting_Size : constant := Word_Size;
  for Adjustable_Lighting'size use
     Adjustable_Lighting_Size;

  for Lighting_Adjustment use
   record
     Lighting_Element
        at 0 range 0..Adjustable_Lighting_Size-1;
     Intensity
        at Adjustable_Lighting_Size/Bytes range 0..Word_Size-1;
  end record;
```

```
for Lighting_Adjustment'size use
   Adjustable_Lighting_Size + Word_Size;

Environmental_Set_Size : constant := Word_Size;
for Environmental_Set'size use
   Environmental_Set_Size;

for Environment_Adjustment use
 record
   ID
      at 0 range 0..Environmental_Set_Size-1;
   Intensity
      at Environmental_Set_Size/Bytes range 0..Word_Size-1;
end record;
for Environment_Adjustment'size use
   Environmental_Set_Size + Word_Size;

Visual_Range_Size : constant := Word_Size;
for Visual_Range'size use Visual_Range_Size;

for Visual_Range_Adjustment use
 record
   ID
      at 0 range 0..Visual_Range_Size-1;
   Range_Set
      at Visual_Range_Size/Bytes range 0..Word_Size-1;
end record;
for Visual_Range_Adjustment'size use
   Visual_Range_Size + Word_Size;

Cloud_Adjustment_Size : constant := Word_Size;
for Cloud_Adjustment'size use Cloud_Adjustment_Size;

for Cloud_Level_Adjustment use
 record
   Cloud_Level
      at 0 range 0..Cloud_Adjustment_Size-1;
   Adjustment_Height
      at Cloud_Adjustment_Size/Bytes range 0..Word_Size-1;
end record;
for Cloud_Level_Adjustment'size use
```

```
   Cloud_Adjustment_Size + Word_Size;

Month_In_A_Year_Size : constant := Word_Size;
for Month_In_A_Year'size use Month_In_A_Year_Size;

for Year_Time use
 record
   Day
      at 0
      range 0..Word_Size-1;
   Month
      at Word_Size/Bytes
      range 0..Month_In_A_Year_Size-1;
   Year
      at Word_Size/Bytes +
         Month_In_A_Year_Size/Bytes
      range 0..Word_Size-1;
end record;
for Year_Time'size use
   Word_Size +
   Month_In_A_Year_Size +
   Word_Size;

-- 10.22.3.7
Emitter_Frequency_Data_Size : constant :=
   Moving_Model_Types.Emitter_Frequency_Data_Size;

for Emitter_Set use
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   Emitter_Data
      at Moving_Model_ID_Size/Bytes
      range 0..Emitter_Frequency_Data_Size-1;
   State
      at Moving_Model_ID_Size/Bytes +
         Emitter_Frequency_Data_Size/Bytes
      range 0..Byte_Size-1;
end record;
for Emitter_Set'size use
```

```
   Moving_Model_ID_Size +
   Emitter_Frequency_Data_Size +
   Byte_Size;

Model_Damage_Data_Size : constant :=
   Moving_Model_Types.Model_Damage_Data_Size;

for Battle_Damage_Set use
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   Desired_Damage
      at Moving_Model_ID_Size/Bytes
      range 0..Model_Damage_Data_Size-1;
end record;
for Battle_Damage_Set'size use
   Moving_Model_ID_Size +
   Model_Damage_Data_Size;

Model_Lighting_Size : constant :=
   Moving_Model_Types.Model_Lighting_Size;

for Model_Lighting_Set use
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   Desired_Lighting
      at Moving_Model_ID_Size/Bytes
      range 0..Model_Lighting_Size-1;
   State
      at Moving_Model_ID_Size/Bytes +
         Model_Lighting_Size/Bytes
      range 0..Byte_Size-1;
end record;
for Model_Lighting_Set'size use
   Moving_Model_ID_Size +
   Model_Lighting_Size +
   Byte_Size;
```

```
Articulated_Part_Data_Size : constant :=
   Moving_Model_Types.Articulated_Part_Data_Size;

for Articulated_Device_Set use
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   Device_Change
      at Moving_Model_ID_Size/Bytes
      range 0..Articulated_Part_Data_Size-1;
end record;
for Articulated_Device_Set'size use
   Moving_Model_ID_Size +
   Articulated_Part_Data_Size;

Weapon_Station_Loading_Size : constant :=
   Global_Message_Types.Weapon_Station_Loading_Size;

for Weapon_Load_Set use
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   Desired_Load
      at Moving_Model_ID_Size/Bytes
      range 0..Weapon_Station_Loading_Size-1;
end record;
for Weapon_Load_Set'size use
   Moving_Model_ID_Size +
   Weapon_Station_Loading_Size;

Station_Weapon_Load_Size : constant :=
   Global_Message_Types.Station_Weapon_Load_Size;

for Weapon_Fire_Command use
 record
   ID
      at 0
      range 0..Moving_Model_ID_Size-1;
   Fired_Weapon
```

```
        at Moving_Model_ID_Size/Bytes
        range 0..Station_Weapon_Load_Size-1;
  end record;
  for Weapon_Fire_Command'size use
     Moving_Model_ID_Size +
     Station_Weapon_Load_Size;

  for Moving_Model_Complexity_Set use
   record
     ID
        at 0
        range 0..Moving_Model_ID_Size-1;
     Complexity
        at Moving_Model_ID_Size/Bytes
        range 0..Word_Size-1;
  end record;
  for Moving_Model_Complexity_Set'size use
     Moving_Model_ID_Size + Word_Size;

--*
end IOS_Output_Interface_Types;
```

```
-------------------------------------------------------------------------
-- %Z% Unit Name:        IOS_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:   12 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:         %I%
-- %Z% Delta Date:       %G%
-- %Z% Current Release: %R%
-------------------------------------------------------------------------
--
-- Purpose:
--  This package specifies all the message objects which are sent by
--  the IOS segment.
--
-- Adaptation:
--  The first step in adaptation is to determine which of the
--  functions in this segment will not be performed, based on simula
--  tor requirements.  The messages associated with these functions
--  need not be sent, andshould therefore be deleted or commented
--  out.
--
--  Each message declaration is followed by a comment line containing
--  "Destination:" and the abbreviations of the segment(s) which
--  receivethis message.  These comments should be modified to
--  account for
--  (a) the presence or absence of other segments, and (b) the
--  requirementsof the other segments for data.  For example, if
--  segment X is absent, then the notation that segment X is a
--  destination of a givenmessage should be removed.  Similarly, when
--  segment Y does not requirethe data in a given message, then the
--  notation that segment Y is adestination for that message should
--  be removed.
--
--  When the segment abbreviations have all been removed for a
--  message, it is clear that this message need not be sent, and the
--  message object declaration itself may be commented out or
--  deleted.
--
```

```
with IOS_Output_Interface_Types;
with Engineering_Units;
with Control_Types;
with Global_Message_Types;
with Moving_Model_Types;
--
package IOS_Output_Interface is
--
--*************************************************************
--*                                                           *
--/ 10.23   IOS Output Interface                              *
--*                                                           *
--*************************************************************


--***************Function:
--*
--/ 10.23.1   Simulator Control
--*

--SEND-ON-CHANGE OUTPUTS


Simulation_State_Command :
   IOS_Output_Interface_Types.
   Segment_Simulation_State_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Training_Mode_Command :
   IOS_Output_Interface_Types.
   Segment_Training_Mode_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Quick_Action_Task_Message :
   IOS_Output_Interface_Types.
   Quick_Action_Task_Command;
--
```

```
-- Destination: PRO,NAV
--

Run_Mode_Freeze_Message :
   IOS_Output_Interface_Types.
   Run_Mode_Freeze_Command;
--
-- Destination: FD,FS
--

Visual_Eyepoint_Active :
   IOS_Output_Interface_Types.
   Visual_Eyepoint;
--
-- Destination: VIS
--

Simulator_Control_Discrete_Message :
   IOS_Output_Interface_Types.
   IOS_Discrete_Output_And_State;
--
-- Destination: ENV, FD,FC,FS,VIS,PHC,RDR,NAV,WPN,EW,PRO
--

Performance_Test_Request_Message :
   IOS_Output_Interface_Types.
   Performance_Test_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Snapshot_Function_Message :
   IOS_Output_Interface_Types.
   Snapshot_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Record_Request_Message :
   IOS_Output_Interface_Types.
   Record_Command;
```

```
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Playback_Request_Message :
   IOS_Output_Interface_Types.
   Playback_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Off_Line_Diagnostic_Message :
   IOS_Output_Interface_Types.
   Off_Line_Diagnostic_Command;

--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Remote_Controlled_Diagnostics_Message :
   IOS_Output_Interface_Types.
   Remote_Controlled_Diagnostics_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

On_Line_Diagnostic_Message :
   IOS_Output_Interface_Types.
   On_Line_Diagnostic_Command;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

Time_Change_Message :
   IOS_Output_Interface_Types.
   Time_Request;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR
--

Multiple_Simulator_Environment_Message :
```

```
   IOS_Output_Interface_Types.
   Multiple_Simulator_Environment_Command;
--
-- Destination: ENV
--

--***************Function:
--*
--/ 10.23.2   Ownship Status and Control
--*

--ITERATIVE OUTPUTS

Clock_Tick_Message_Max_Rate :
   IOS_Output_Interface_Types.
   Clock_Tick_Messages;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,VIS,RDR,PHC
--

--SEND-ON-CHANGE OUTPUTS


Parameter_Slew_Message :
   IOS_Output_Interface_Types.
   Parameter_Slew_Demand;
--
-- Destination: FD
--

Add_Cargo_Message :
   IOS_Output_Interface_Types.
   Add_Cargo_Command;
--
-- Destination: FD
--

External_Connection_Message :
   IOS_Output_Interface_Types.
   External_Connection_Command;
--
```

```
-- Destination: FD,FS,PHC
--

Parameter_Change_Message :
   IOS_Output_Interface_Types.
   Parameter_Change_Request;
--
-- Destination: FD,PRO,VIS,FS
--

Hydraulic_Fluid_Quantity_Change_Message :
   IOS_Output_Interface_Types.
   Hydraulic_Fluid_Quantity_Change;
--
-- Destination: FS
--

Fuel_Quantity_Change_Message :
   IOS_Output_Interface_Types.
   Fuel_Quantity_Change;
--
-- Destination: FS
--

Reset_Message :
   IOS_Output_Interface_Types.
   Reset_Demand;
--
-- Destination: FS,PRO,FC,WPN
--

Icing_Level_Adjustment_Message :
   IOS_Output_Interface_Types.
   Icing_Level_Adjustment;
--
-- Destination: ENV,FD,PRO
--

Engine_Oil_Quantity_Adjustment_Message :
   IOS_Output_Interface_Types.
   Engine_Oil_Quantity_Adjustment;
```

```
--
-- Destination: PRO
--

IOS_Discrete_Request_Message :
   IOS_Output_Interface_Types.
   IOS_Discrete_Command_And_State;
--
-- Destination: FD,NAV,PHC,FC
--



--***************Function:
--*
--/ 10.23.3   Ownship Malfunction
--*

--SEND-ON-CHANGE OUTPUTS


Malfunction_Message :
   Control_Types.
   Malfunction_Demand;
--
-- Destination: ENV, FD,FC,FS,PRO,NAV,EW,WPN,RDR
--


--***************Function:
--*
--/ 10.23.4   Ownship Controls Disagreement
--*

--NONE


--***************Function:
--*
--/ 10.23.5   Navigation/Communication Status and Control
--*
```

```
--SEND-ON-CHANGE OUTPUTS

Waypoint_Change_Message :
   Global_Message_Types.
   Waypoint_Change;
--
-- Destination: NAV
--

Platform_Radio_Set_Message :
   IOS_Output_Interface_Types.
   Platform_Radio_Set;
--
-- Destination: NAV
--

Training_Area_Boundary_Message :
   IOS_Output_Interface_Types.
   Training_Area_Boundaries;
--
-- Destination: ENV,NAV,EW,WPN,VIS,RDR
--

Ownship_Position_Change_Demand :
   Engineering_Units.
   Earth_Position_Components;
--
-- Destination: ENV,FD,NAV,EW,VIS,RDR
--


--***************Function:
--*
--/ 10.23.6   Physical/Natural Environment Status and Control
--*


--SEND-ON-CHANGE OUTPUTS
```

```
Wind_Shear_Message :
   IOS_Output_Interface_Types.
   Wind_Shear;
--
-- Destination: ENV
--

Wind_Direction_Set_Message :
   IOS_Output_Interface_Types.
   Wind_Direction_Set;
--
-- Destination: ENV
--

Wind_Speed_Set_Message :
   IOS_Output_Interface_Types.
   Wind_Speed_Set;
--
-- Destination: ENV
--

Wind_Position_Message :
   IOS_Output_Interface_Types.
   Wind_Position;
--
-- Destination: ENV
--

Wind_Intensity_Message :
   IOS_Output_Interface_Types.
   Wind_Intensity;
--
-- Destination: ENV
--

Turbulence_Demand_Message :
   IOS_Output_Interface_Types.
   Turbulence_Demand;
--
-- Destination: ENV,PHC
--
```

```
Runway_Surface_Condition_Message :
   IOS_Output_Interface_Types.
   Runway_Surface_Condition;
--
-- Destination: ENV,FD,VIS
--

Runway_Condition_Reading_Message :
   IOS_Output_Interface_Types.
   Runway_Condition_Reading;
--
-- Destination: ENV,FD,VIS
--

Pressure_Demand_Message :
   IOS_Output_Interface_Types.
   Pressure_Demand;
--
-- Destination: ENV
--

Temperature_Demand_Message :
   IOS_Output_Interface_Types.
   Temperature_Demand;
--
-- Destination: ENV
--

Visual_Model_Database_Message :
   IOS_Output_Interface_Types.
   Visual_Model_Database;
--
-- Destination: ENV,VIS
--

Thunderstorm_Dynamic_Data_Message :
   Moving_Model_Types.
   Moving_Model_Dynamic_Data;
--
-- Destination: VIS,RDR,ENV,PHC
```

```
--
Thunderstorm_Intensity_Message :
   IOS_Output_Interface_Types.
   Thunderstorm_Intensity_Set;
--
-- Destination: ENV,VIS,RDR,PHC
--

Lighting_Adjustment_Message :
   IOS_Output_Interface_Types.
   Lighting_Adjustment;
--
-- Destination: ENV,NAV,VIS
--

Environmental_Adjustment_Message :
   IOS_Output_Interface_Types.
   Environment_Adjustment;
--
-- Destination: ENV, NAV,VIS,RDR,PHC,FD
--

Visual_Range_Adjustment_Message :
   IOS_Output_Interface_Types.
   Visual_Range_Adjustment;
--
-- Destination: ENV,VIS
--

Cloud_Level_Adjustment_Message :
   IOS_Output_Interface_Types.
   Cloud_Level_Adjustment;
--
-- Destination: ENV,VIS
--

Time_Of_Year_Message :
   IOS_Output_Interface_Types.
   Year_Time;
--
```

```
-- Destination: NAV,VIS
--

Temperature_Profile_Message :
   IOS_Output_Interface_Types.
   Temperature_Profile;
--
-- Destination: ENV
--

Temperature_Lapse_Rate_Message :
   IOS_Output_Interface_Types.
   Temperature_Lapse_Rate;
--
-- Destination: ENV
--

Wind_Speed_Lapse_Rate_Message :
   IOS_Output_Interface_Types.
   Wind_Speed_Lapse_Rate;
--
-- Destination: ENV
--

Wind_Direction_Lapse_Rate_Message :
   IOS_Output_Interface_Types.
   Wind_Direction_Lapse_Rate;
--
-- Destination: ENV
--


--***************Function:
--*
--/ 10.23.7   Tactical and Mission Environment Status and Control
--*

--SEND-ON-CHANGE OUTPUTS


Moving_Model_Dynamic_Data_Message :
```

```
   Moving_Model_Types.
   Moving_Model_Dynamic_Data;
--
-- Destination: ENV, EW,NAV,RDR,VIS,WPN
--

Moving_Model_Deactivation_Message :
   Moving_Model_Types.
   Moving_Model_Deactivation;
--
-- Destination: ENV, EW,NAV,RDR,VIS,WPN
--

Chaff_Creation_Message :
   Moving_Model_Types.
   Chaff_Moving_Model_Unique_Data;
--
-- Destination: ENV,EW,NAV,RDR,VIS,WPN
--

Flare_Creation_Message :
   Moving_Model_Types.
   Flare_Moving_Model_Unique_Data;
--
-- Destination: ENV,EW,NAV,RDR,VIS,WPN
--

Decoy_Creation_Message :
   Moving_Model_Types.
   Decoy_Moving_Model_Unique_Data;
--
-- Destination: ENV,EW,NAV,RDR,VIS,WPN
--

Platform_Creation_Message :
   Moving_Model_Types.
   Platform_Moving_Model_Unique_Data;
--
-- Destination: ENV,EW,NAV,RDR,VIS,WPN
--
```

```
Tanker_Creation_Message :
   Moving_Model_Types.
   Tanker_Moving_Model_Unique_Data;
--
-- Destination: ENV
--

Emitter_Set_Message :
   IOS_Output_Interface_Types.
   Emitter_Set;
--
-- Destination: ENV,EW,NAV,RDR
--

Battle_Damage_Set_Message :
   IOS_Output_Interface_Types.
   Battle_Damage_Set;
--
-- Destination: ENV,EW,FD,RDR,VIS,WPN
--

Model_Lighting_Set_Message :
   IOS_Output_Interface_Types.
   Model_Lighting_Set;
--
-- Destination: ENV,VIS
--

Articulated_Device_Set_Message :
   IOS_Output_Interface_Types.
   Articulated_Device_Set;
--
-- Destination: ENV,RDR,VIS
--

Weapon_Load_Set_Message :
   IOS_Output_Interface_Types.
   Weapon_Load_Set;
--
-- Destination: EW,FD,VIS,WPN
--
```

```
Weapon_Fire_Command_Message :
   IOS_Output_Interface_Types.
   Weapon_Fire_Command;
--
-- Destination: EW,VIS,WPN
--

Moving_Model_Complexity_Set_Message :
   IOS_Output_Interface_Types.
   Moving_Model_Complexity_Set;
--
-- Destination: ENV
--

IFF_Set_Message :
   Moving_Model_Types.
   Moving_Model_IFF_Data;
--
-- Destination: EW,NAV,RDR
--


--***************Function:
--*
--/ 10.23.8   Crew Station Performance Monitoring and Measurement
--*

--NONE


--***************Function:
--*
--/ 10.23.9   Instructor/Operator Station Segment Support
--*

--NONE


--*
end IOS_Output_Interface;
```

```
-----------------------------------------------------------------------
-- %Z% Unit Name:        Environment_Output_Interface_Types
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:        Package Spec (no body)
-- %Z% Unit ID:          (tbd)
-- %Z% Author:           Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  3 August 1993
-- %Z% SCCS Filename:   %M%
-- %Z% Delta ID:        %I%
-- %Z% Delta Date:      %G%
-- %Z% Current Release: %R%
-----------------------------------------------------------------------
--
-- Purpose:
--  This package specifies types for messages which are output only
--  by the Environment segment.  Other packages
--  that include types that may be sent by this segment include
--  Control_Types, Moving_Model_Types, Global_Message_Types and
--  Service_Function_Types.
--
-- Adaptation:
--  The section containing the aircraft/simulator specific types must be
--  modified to match the requirements of the aircraft being simulated
or
--  the requirements for the simulator.  As a general rule, the contents
--  of the section containing reusable types will not need to be
--  modified.  The representation specs in the private part are designed
--  to require little or no modification.
--
with Base_Types;
with Engineering_Units;
with Global_Message_Types;
with Moving_Model_Types;
--
package Environment_Output_Interface_Types is
--
--*********************************************************************
--*                                                                   *
--/ 10.24  Environment Output Interface Types                         *
--*                                                                   *
--*********************************************************************
```

```
--**************************
--*
--/ 10.24.1   Aircraft/Simulator Specific Environment Types
--*
--**************************

-- These constants are important only when you gather, e.g., nav sites
-- together into one message, rather than sending one message for each
-- object.
Maximum_Number_Of_Sites                : constant := 25;
Maximum_Number_Of_Airports             : constant := 1;
Maximum_Number_Of_Runways_Per_Airport  : constant := 2;
Maximum_Number_Of_Threat_Weapons       : constant := 10;
Maximum_Number_Of_Threat_Platforms     : constant := 10;
Maximum_Number_Of_Companion_Vehicles   : constant := 10;

-- Types of sites in the nav data base
type Site_Type is (
   TACAN,
   VORTAC,
   DME,
   ILS,
   IFF,
   Marker_Beacon,
   ADF);

-- Points on this aircraft where a collision might take place
type Collision_Point is (
   Boom_Tip,
   Left_Landing_Gear,
   Nose_Landing_Gear,
   Right_Landing_Gear);

-- Types of precipitation selectable by the instructor
type Precipitation_Type is (
   None,
   Rain,
   Sleet,
   Snow,
   Hail);
```

```
--*****************************************
--*
--/ 10.24.2   Aircraft/Simulator Reusable Environment Types
--*
--*****************************************

subtype Site_Count is Base_Types.Signed_Integer_32
   range 1..Maximum_Number_Of_Sites;
subtype Runway_Count is Base_Types.Signed_Integer_32
   range 1..Maximum_Number_Of_Runways_Per_Airport;
subtype Threat_Weapon_Count is Base_Types.Signed_Integer_32
   range 1..Maximum_Number_Of_Threat_Weapons;
subtype Threat_Platform_Count is Base_Types.Signed_Integer_32
   range 1..Maximum_Number_Of_Threat_Platforms;
subtype Companion_Vehicle_Count is Base_Types.Signed_Integer_32
   range 1..Maximum_Number_Of_Companion_Vehicles;

-- Runway data
type Runway_Data is
  record
    Runway_Heading  : Engineering_Units.Degrees;
                      -- RDR, VIS, IOS, FD
    Runway_Location : Engineering_Units.Earth_Position_Components;
                      -- RDR, VIS, IOS, FD
    Runway_Gradient : Engineering_Units.Inches_Per_Foot;
                      -- RDR, VIS, IOS, FD
    Runway_Width    : Engineering_Units.Feet;
                      -- RDR, VIS, IOS, FD
    Runway_Length   : Engineering_Units.Feet;
                      -- RDR, VIS, IOS, FD
 end record;

type Airport_Runways is array (Runway_Count) of Runway_Data;


-- Threat weapons
type Threat_Weapon_Dynamic_Data_Array is array (Threat_Weapon_Count) of
   Moving_Model_Types.Moving_Model_Dynamic_Data;
```

```
-- External entities
type Threat_Platform_Dynamic_Data_Array is array (Threat_Platform_Count)
of
   Moving_Model_Types.Moving_Model_Dynamic_Data;

type Companion_Vehicle_Dynamic_Data_Array is array
(Companion_Vehicle_Count) of
   Moving_Model_Types.Moving_Model_Dynamic_Data;

-- Collision
type Collision_Status is (Collision, No_Collision);

type Collisions is (Terrain, Moving_Model);

-- Weather
type Precipitation_Components is
 record
   Precipitation_Kind : Precipitation_Type;
   Intensity          : Engineering_Units.Zero_To_Ten;
 end record;


--*****************************************
--*
--/ 10.24.3   Environment Segment Output Records
--*
--*****************************************


--***************Function:
--*
--/ 10.24.3.1   MSE Interaction
--*

--NONE


--***************Function:
--*
--/ 10.24.3.2   Atmosphere
--*
type Atmosphere_Quarter_Rate is
```

```
 record
   Air_Density_Ratio                 : Base_Types.Float_32;
                                       --NAV
   Ambient_Air_Pressure              : Engineering_Units.PSI;
                                       --NAV,  PRO
   Ambient_Air_Temperature           : Engineering_Units.Degrees_C;
                                       --PRO, NAV
   Dynamic_Pressure                  : Engineering_Units.PSI;
                                       --FC, NAV
   Impact_Pressure                   : Engineering_Units.PSI;
                                       --NAV,FC
   Pressure_Altitude                 : Engineering_Units.Feet;
                                       --PHC,PRO,NAV,FC,IOS
   Sea_Level_Barometric_Pressure     : Engineering_Units.Inches_Hg;
                                       --NAV
   Static_Pressure_Ratio             : Base_Types.Float_32;
                                       --NAV
   Total_Air_Pressure                : Engineering_Units.PSI;
                                       --NAV
 end record;

type Weather_Quarter_Rate is
 record
  Wind_Angular_Velocity    : Engineering_Units.Angular_Velocity_Components;
                                  -- FD
  Wind_Earth_Axis_Velocity : Engineering_Units.Earth_Velocity_Components;
                                  -- FD
   Precipitation             : Precipitation_Components;
                                  -- FD
 end record;


--***************Function:
--*
--/ 10.24.3.3   Ownship Weapons' Damage Assessment
--*

-- See Moving_Model_Types for declarations of
-- Scoring_Damage_Data and Scoring_Activation_Status
```

```
--***************Function:
--*
--/ 10.24.3.4   Threat Weapon Dynamics
--*

type Threat_Weapon_Dynamics_Half_Rate is
 record
   Number_Of_Threat_Weapons     : Threat_Weapon_Count;
   Threat_Weapons_Dynamic_Data : Threat_Weapon_Dynamic_Data_Array;
                                    --RDR, VIS, IOS
 end record;

type Platform_Stores_Data is
 record
   ID           : Moving_Model_Types.Moving_Model_ID;          --EW
   Station      : Global_Message_Types.Stores_Station;         --EW
   Stores_Type : Global_Message_Types.Station_Weapon_Load;   --EW
 end record;

type Platform_Weapon_Fire_Status is
 record
   Weapon_Fired_From : Moving_Model_Types.Moving_Model_ID;  -- PHC, IOS,
EW
   Weapon_Fired       : Moving_Model_Types.Moving_Model_ID;  -- PHC, IOS
   Intended_Target    : Moving_Model_Types.Moving_Model_ID;  -- EW
 end record;


--***************Function:
--*
--/ 10.24.3.5   External Entity
--*

type Threat_Platform_Dynamics_Half_Rate is
 record
   Number_Of_Threat_Platforms     : Threat_Platform_Count;
   Threat_Platform_Dynamic_Data : Threat_Platform_Dynamic_Data_Array;
                                    --RDR, VIS, IOS
 end record;

type Companion_Vehicles_Half_Rate is
 record
```

```
  Number_Of_Companion_Vehicles    : Companion_Vehicle_Count;
  Companion_Vehicle_Dynamic_Data : Companion_Vehicle_Dynamic_Data_Array;
 end record;

-- See Moving_Model_Types for declarations of
-- Platform_Moving_Model_Unique_Data, Tanker_Moving_Model_Unique_Data,
-- and Moving_Model_Deactivation


--***************Function:
--*
--/ 10.24.3.6   External Entity's Chaff and Flares
--*

-- See Moving_Model_Types for declarations of
-- Chaff_And_Flares_Moving_Model_Data and Chaff_And_Flares_Detail_Data


--***************Function:
--*
--/ 10.24.3.7    Database Management
--*

--NONE


--***************Function:
--*
--/ 10.24.3.8    Threat Environment Database
--*

--NONE


--***************Function:
--*
--/ 10.24.3.9    Navigation Environment
--*

-- SEND_ON_CHANGE ONLY
```

```
--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Site_Data is
 record
  ID       : Site_Type;                                    -- RDR, VIS
  Location : Engineering_Units.Earth_Position_Components;  -- RDR, VIS
 end record;

--
-- ...or it will collect them into an array...
--
type Site_Set is array (Site_Count) of Site_Data;

--
-- ...and send the ones which have changed in one of these messages:
--
type NAV_ENV_Site_Data is
 record
   Number_Of_Sites        : Site_Count;
   Station_Site_Locations : Site_Set;
 end record;

--
-- Depending on the implementation, this segment will either send a
-- number of these messages:
--
type Airport_Runway_Data is
 record
   Number_Of_Runways : Runway_Count;
   Configuration     : Airport_Runways;
 end record;

--
-- ...or it will collect them into an array...
--
subtype Airport_Count is Base_Types.Signed_Integer_32
   range 1..Maximum_Number_Of_Airports;

type Airports_Data is array (Airport_Count) of Airport_Runway_Data;
```

```
--
-- ...and send the ones which have changed in one of these messages:
--
type Airports is
 record
   Number_Of_Airports   : Airport_Count;
   Runways_At_Airports  : Airports_Data;
 end record;


--***************Function:
--*
--/ 10.24.3.10   Collision Detection
--*

--SEND-ON-CHANGE ONLY

type Collision_Data is
 record
   Collision_Point_ID        : Collision_Point;
  Collision_Point_Position : Engineering_Units.Earth_Position_Components;
   Current_Collision_Status : Collision_Status;
   Collision_Kind            : Collisions;
 end record;



--***************Function:
--*
--/ 10.24.3.11   Radar Database / Gaming Area
--*

-- No types currently defined.  When they are defined, they should go
-- into Service_Function_Types.


--***************Function:
--*
--/ 10.24.3.12   Visual Database / Gaming Area
--*
```

```
-- No types currently defined.  When they are defined, they should go
-- into Service_Function_Types.


--***************Function:
--*
--/ 10.24.3.13  Spatial Relations
--*

-- Spatial Relations is a service function, which can be performed by
-- ENV, RDR or VIS.  Definitions for service functions belong in
-- Service_Function_Types.


--***************Function:
--*
--/ 10.24.3.14  Occulting
--*

-- Occulting is a service function, which can be performed by
-- ENV, RDR or VIS.  Definitions for service functions belong in
-- Service_Function_Types.


--***************Function:
--*
--/ 10.24.3.15  Environment Support
--*

-- See Control_Types for responses to IOS


--*************************
--*
--/ 10.24.4   Environment Representation Specs
--*
--*************************
private

   --
```

```
-- Declarations to make representation specs more readable
--
Bytes     : constant := 8; -- Bits per byte
Byte_Size : constant := 1 * Bytes;
Word_Size : constant := 4 * Bytes;

Earth_Position_Components_Size   : constant :=
   Engineering_Units.Earth_Position_Components_Size;
Angular_Velocity_Components_Size : constant :=
   Engineering_Units.Angular_Velocity_Components_Size;
Earth_Velocity_Components_Size   : constant :=
   Engineering_Units.Earth_Velocity_Components_Size;
Moving_Model_ID_Size             : constant :=
   Moving_Model_Types.Moving_Model_ID_Size;

-- 10.24.3.2
for Atmosphere_Quarter_Rate use
 record
   Air_Density_Ratio              at 0                     range
0..Word_Size-1;
   Ambient_Air_Pressure           at 1 * Word_Size/Bytes range
0..Word_Size-1;
   Ambient_Air_Temperature        at 2 * Word_Size/Bytes range
0..Word_Size-1;
   Dynamic_Pressure               at 3 * Word_Size/Bytes range
0..Word_Size-1;
   Impact_Pressure                at 4 * Word_Size/Bytes range
0..Word_Size-1;
   Pressure_Altitude              at 5 * Word_Size/Bytes range
0..Word_Size-1;
   Sea_Level_Barometric_Pressure at 6 * Word_Size/Bytes range
0..Word_Size-1;
   Static_Pressure_Ratio          at 7 * Word_Size/Bytes range
0..Word_Size-1;
   Total_Air_Pressure             at 8 * Word_Size/Bytes range
0..Word_Size-1;
 end record;
for Atmosphere_Quarter_Rate'size use 9 * Word_Size;

Precipitation_Type_Size : constant := 4 * Bytes;
for Precipitation_Type'size use Precipitation_Type_Size;

for Precipitation_Components use
```

```
 record
   Precipitation_Kind at 0               range
0..Precipitation_Type_Size-1;
   Intensity          at Word_Size/Bytes range 0..Word_Size-1;
 end record;
Precipitation_Components_Size : constant :=
   Precipitation_Type_Size + Word_Size;
for Precipitation_Components'size use Precipitation_Components_Size;

for Weather_Quarter_Rate use
 record
   Wind_Angular_Velocity    at  0
                            range 0..
                               Angular_Velocity_Components_Size-1;
  Wind_Earth_Axis_Velocity at Angular_Velocity_Components_Size/Bytes
                            range 0..
                               Earth_Velocity_Components_Size-1;
   Precipitation            at
                              Angular_Velocity_Components_Size/Bytes
+
                               Earth_Velocity_Components_Size/Bytes
                           range 0..Precipitation_Components_Size-1;
 end record;
for Weather_Quarter_Rate'size use
   Angular_Velocity_Components_Size +
   Earth_Velocity_Components_Size +
   Precipitation_Components_Size;

-- 10.24.3.4
Threat_Weapon_Dynamic_Data_Array_Size : constant :=
   Moving_Model_Types.Moving_Model_Dynamic_Data_Size *
   Maximum_Number_Of_Threat_Weapons;
for Threat_Weapon_Dynamic_Data_Array'size use
   Threat_Weapon_Dynamic_Data_Array_Size;

for Threat_Weapon_Dynamics_Half_Rate use
 record
  Number_Of_Threat_Weapons    at 0              range 0..Word_Size-1;
   Threat_Weapons_Dynamic_Data at Word_Size/Bytes range 0..
      Threat_Weapon_Dynamic_Data_Array_Size-1;
 end record;
for Threat_Weapon_Dynamics_Half_Rate'size use
```

```
   Word_Size +
   Threat_Weapon_Dynamic_Data_Array_Size;

for Platform_Stores_Data use
 record
   ID           at 0
                range 0..Moving_Model_ID_Size-1;
   Station      at Moving_Model_ID_Size/Bytes
                range 0..Byte_Size-1;
   Stores_Type  at Moving_Model_ID_Size/Bytes + 1
                range 0..Byte_Size-1;
 end record;
for Platform_Stores_Data'size use
   Moving_Model_ID_Size + (2 * Byte_Size);

for Platform_Weapon_Fire_Status use
 record
   Weapon_Fired_From at  0
                      range 0..Moving_Model_ID_Size-1;
   Weapon_Fired       at  Moving_Model_ID_Size/Bytes
                      range 0..Moving_Model_ID_Size-1;
   Intended_Target    at 2 * Moving_Model_ID_Size/Bytes
                      range 0..Moving_Model_ID_Size-1;
 end record;
for Platform_Weapon_Fire_Status'size use 3 * Moving_Model_ID_Size;

-- 10.24.3.5
Threat_Platform_Dynamic_Data_Array_Size : constant :=
   Moving_Model_Types.Moving_Model_Dynamic_Data_Size *
   Maximum_Number_Of_Threat_Platforms;
for Threat_Platform_Dynamic_Data_Array'size use
   Threat_Platform_Dynamic_Data_Array_Size;

for Threat_Platform_Dynamics_Half_Rate use
 record
   Number_Of_Threat_Platforms    at 0
                                 range 0..Word_Size-1;
   Threat_Platform_Dynamic_Data at Word_Size/Bytes
                                 range 0..
                           Threat_Platform_Dynamic_Data_Array_Size-1;
 end record;
```

```
for Threat_Platform_Dynamics_Half_Rate'size use
   Word_Size + Threat_Platform_Dynamic_Data_Array_Size;

Companion_Vehicle_Dynamic_Data_Array_Size : constant :=
   Moving_Model_Types.Moving_Model_Dynamic_Data_Size *
   Maximum_Number_Of_Companion_Vehicles;
for Companion_Vehicle_Dynamic_Data_Array'size use
   Companion_Vehicle_Dynamic_Data_Array_Size;

for Companion_Vehicles_Half_Rate use
 record
   Number_Of_Companion_Vehicles   at 0
                                  range 0..Word_Size-1;
   Companion_Vehicle_Dynamic_Data at Word_Size/Bytes
                                  range 0..
Companion_Vehicle_Dynamic_Data_Array_Size-1;
 end record;
for Companion_Vehicles_Half_Rate'size use
   Word_Size + Companion_Vehicle_Dynamic_Data_Array_Size;

-- 10.24.3.9
Site_Type_Size : constant := 4 * Bytes;
for Site_Type'size use Site_Type_Size;

for Site_Data use
 record
   ID       at 0 range 0..Site_Type_Size-1;
  Location at Word_Size/Bytes range 0..Earth_Position_Components_Size-1;
 end record;
Site_Data_Size : constant :=
   Site_Type_Size + Earth_Position_Components_Size;
for Site_Data'size use Site_Data_Size;

Site_Set_Size : constant :=
   Site_Data_Size * Maximum_Number_Of_Sites;
for Site_Set'size use Site_Set_Size;

for NAV_ENV_Site_Data use
 record
   Number_Of_Sites        at 0 range 0..Word_Size-1;
   Station_Site_Locations at 4 range 0..Site_Set_Size-1;
```

```
 end record;
for NAV_ENV_Site_Data'size use Word_Size + Site_Set_Size;

for Runway_Data use
 record
   Runway_Heading  at  0
                   range 0..Word_Size-1;
   Runway_Location at Word_Size/Bytes
                   range 0..Earth_Position_Components_Size-1;
   Runway_Gradient at
                      Word_Size/Bytes +
                      Earth_Position_Components_Size/Bytes
                   range 0..Word_Size-1;
   Runway_Width    at
                      Word_Size/Bytes +
                      Earth_Position_Components_Size/Bytes +
                      Word_Size/Bytes
                   range 0..Word_Size-1;
   Runway_Length   at
                      Word_Size/Bytes +
                      Earth_Position_Components_Size/Bytes +
                      Word_Size/Bytes +
                      Word_Size/Bytes
                   range 0..Word_Size-1;
 end record;
Runway_Data_Size : constant :=
   Word_Size +
   Earth_Position_Components_Size +
   Word_Size +
   Word_Size +
   Word_Size;
for Runway_Data'size use Runway_Data_Size;

Airport_Runways_Size : constant :=
   Runway_Data_Size * Maximum_Number_Of_Runways_Per_Airport;
for Airport_Runways'size use Airport_Runways_Size;

for Airport_Runway_Data use
 record
   Number_Of_Runways at 0 range 0..Word_Size-1;
   Configuration     at 4 range 0..Airport_Runways_Size-1;
```

```
 end record;
Airport_Runway_Data_Size : constant :=
   Word_Size + Airport_Runways_Size;
for Airport_Runway_Data'size use Airport_Runway_Data_Size;

Airports_Data_Size : constant :=
   Airport_Runway_Data_Size * Maximum_Number_Of_Airports;

for Airports_Data'size use Airports_Data_Size;

for Airports use
 record
   Number_Of_Airports  at 0 range 0..(4 * Bytes)-1;
   Runways_At_Airports at 4 range 0..Airport_Runway_Data_Size-1;
 end record;
for Airports'size use Word_Size + Airport_Runway_Data_Size;

-- 10.24.3.10
Collision_Point_Size : constant := 4 * Bytes;
for Collision_Point'size use Collision_Point_Size;

Collision_Status_Size : constant := 8;
for Collision_Status'size use Collision_Status_Size;

Collisions_Size : constant := 8;
for Collisions'size use Collisions_Size;

for Collision_Data use
 record
   Collision_Point_ID       at  0
                            range 0..Collision_Point_Size-1;
   Collision_Point_Position at  Collision_Point_Size/Bytes
                         range 0..Earth_Position_Components_Size-1;
   Current_Collision_Status at Collision_Point_Size/Bytes +
                               Earth_Position_Components_Size/Bytes
                            range 0..Collision_Status_Size-1;
   Collision_Kind           at Collision_Point_Size/Bytes +
                               Earth_Position_Components_Size/Bytes
+ 1
                            range 0..Collisions_Size-1;
 end record;
for Collision_Data'size use
```

```
        Collision_Point_Size +
        Earth_Position_Components_Size +
        Collision_Status_Size +
        Collisions_Size;

end Environment_Output_Interface_Types;
```

```
--------------------------------------------------------------------
-- %Z% Unit Name:          Environment_Output_Interface
-- %Z% Source Pathname: %P%
-- %Z% Unit Type:          Package Spec (no body)
-- %Z% Unit ID:            (tbd)
-- %Z% Author:             Gary Kamsickas, Bob Crispen, et al.
-- %Z% Date of Origin:  12 August 1993
-- %Z% SCCS Filename:    %M%
-- %Z% Delta ID:           %I%
-- %Z% Delta Date:         %G%
-- %Z% Current Release: %R%
--------------------------------------------------------------------
--
-- Purpose:
--   This package specifies all the message objects which are sent by the
--   Environment segment.
--
-- Adaptation:
--   The first step in adaptation is to determine which of the functions
--   in this segment will not be performed, based on simulator
requirements.
--   The messages associated with these functions need not be sent, and
--   should therefore be deleted or commented out.
--
--   Each message declaration is followed by a comment line containing
--   "Destination:" and the abbreviations of the segment(s) which receive
--   this message.  These comments should be modified to account for
--   (a) the presence or absence of other segments, and (b) the
requirements
--  of the other segments for data.  For example, if segment X is absent,
--   then the notation that segment X is a destination of a given
--   message should be removed.  Similarly, when segment Y does not
require
--   the data in a given message, then the notation that segment Y is a
--   destination for that message should be removed.
--
--   When the segment abbreviations have all been removed for a message,
--   it is clear that this message need not be sent, and the message
--   object declaration itself may be commented out or deleted.
--
--  The four service functions: Radar Database, Visual Database, Spatial
--  Relations and Occulting have messages which must each be sent by one
```

```
--  and only one segment.  Only one of the following three segments:
--  Environment, Radar or Visual, may send these messages.  Modify the
--  output interface packages for each of these three segments in
--  accordance with the assignment of the functions to the segments,
--  commenting or uncommenting declarations accordingly.
--
with Environment_Output_Interface_Types;
with Control_Types;
with Service_Function_Types;
with Moving_Model_Types;
--
package Environment_Output_Interface is
--
--***************************************************************
--*                                                             *
--/ 10.25  Environment Output Interface                         *
--*                                                             *
--***************************************************************


--***************Function:
--*
--/ 10.25.1   MSE Interaction
--*

--NONE


--***************Function:
--*
--/ 10.25.2   Atmosphere
--*

Atmosphere_Quarter_Rate_Outputs :
   Environment_Output_Interface_Types.
   Atmosphere_Quarter_Rate;
--
-- Destination: FD, FC, IOS, NAV, PHC, PRO
--

Weather_Quarter_Rate_Outputs :
```

```
   Environment_Output_Interface_Types.
   Weather_Quarter_Rate;
--
-- Destination: FD
--


--***************Function:
--*
--/ 10.25.3   Ownship Weapons' Damage Assessment
--*

--SEND-ON-CHANGE OUTPUTS

Moving_Model_Damage_Occurrence :
   Moving_Model_Types.
   Scoring_Damage_Data;
--
-- Destination: IOS, EW
--

Moving_Model_Scoring_Activation :
   Moving_Model_Types.
   Scoring_Activation_Status;
--
-- Destination: EW, IOS
--


--***************Function:
--*
--/ 10.25.4   Threat Weapon Dynamics
--*

Threat_Weapon_Dynamics_Half_Rate_Outputs :
   Environment_Output_Interface_Types.
   Threat_Weapon_Dynamics_Half_Rate;
--
-- Destination:  RDR, VIS, IOS
--
```

```
--SEND-ON-CHANGE OUTPUTS

Platform_Fire_Occurrence :
   Environment_Output_Interface_Types.
   Platform_Weapon_Fire_Status;
--
-- Destination: PHC, EW, IOS
--

Platform_Stores_Data_Update :
   Environment_Output_Interface_Types.
   Platform_Stores_Data;
--
-- Destination: EW
--

--***************Function:
--*
--/ 10.25.5   External Entity
--*

Threat_Platform_Dynamics_Half_Rate_Outputs :
   Environment_Output_Interface_Types.
   Threat_Platform_Dynamics_Half_Rate;
--
-- Destination:  EW, IOS, VIS, RDR, WPN, NAV
--

Companion_Vehicles_Half_Rate_Outputs :
   Environment_Output_Interface_Types.
   Companion_Vehicles_Half_Rate;
--
-- Destination: FD, EW,FS,IOS,NAV,RDR,VIS
--

--SEND-ON-CHANGE OUTPUTS

Threat_Platform_Unique_Data :
   Moving_Model_Types.
   Platform_Moving_Model_Unique_Data;
--
```

```
-- Destination:  EW, IOS, VIS, RDR, WPN, NAV, FS
--

Companion_Change_Data :
   Moving_Model_Types.
   Platform_Moving_Model_Unique_Data;
--
-- Destination: FD, EW, FS, IOS, NAV, VIS, RDR, PHC
--

Tanker_Change_Data :
   Moving_Model_Types.
   Tanker_Moving_Model_Unique_Data;
--
-- Destination: FD, FS, IOS, NAV, VIS, RDR, PHC
--

Companion_Tanker_Model_Deactivate :
   Moving_Model_Types.
   Moving_Model_Deactivation;
--
-- Destination: FD, FS, IOS, NAV, VIS, RDR, PHC, EW
--


--***************Function:
--*
--/ 10.25.6   External Entity's Chaff and Flares
--*

External_Chaff_And_Flares_Half_Rate_Outputs :
   Moving_Model_Types.
   Chaff_And_Flares_Moving_Model_Data;
--
-- Destination:  EW, RDR, VIS, IOS
--

External_Chaff_And_Flares_Sixteenth_Rate_Outputs :
   Moving_Model_Types.
   Chaff_And_Flares_Detail_Data;
--
```

```
-- Destination:  EW, RDR, VIS, IOS
--


--***************Function:
--*
--/ 10.25.7    Database Management
--*

--NONE


--***************Function:
--*
--/ 10.25.8    Threat Environment Database
--*

--NONE


--***************Function:
--*
--/ 10.25.9    Navigation Environment
--*

-- SEND-ON-CHANGE OUTPUTS

Airports_Output :
   Environment_Output_Interface_Types.
   Airports;
--
-- Destination: NAV, IOS, RDR, VIS, FD
--

NAV_Aid_Site_Output :
   Environment_Output_Interface_Types.
   NAV_ENV_Site_Data;
--
-- Destination: NAV, RDR, VIS
--
```

```
--***************Function:
--*
--/ 10.25.10   Collision Detection
--*

--SEND-ON-CHANGE OUTPUTS

Collision_Data_Change :
   Environment_Output_Interface_Types.
   Collision_Data;
--
-- Destination: VIS, WPN,FD,IOS
--

--***************Function:
--*
--/ 10.25.11   Radar Database
--*

-- NONE


--***************Function:
--*
--/ 10.25.12   Visual Database
--*

--NONE


--***************Function:
--*
--/ 10.25.13   Spatial Relations
--*

Ownship_Height_Above_Terrain_Max_Rate_Outputs :
   Service_Function_Types.
   Ownship_Height_Above_Terrain;
--
-- Destination: NAV, RDR, WPN, PHC, VIS, FD
```

```
--

Moving_Models_Height_Above_Terrain_Max_Rate_Outputs :
   Service_Function_Types.
   Moving_Models_Height_Above_Terrain;
--
-- Destination: NAV, RDR, WPN, PHC, VIS, FD
--

--SEND-ON-CHANGE OUTPUTS

-- Position_Range_Change :
--    Service_Function_Types.
--    Position_Range_Update;
-- Destination: RDR, NAV, VIS
--
-- Assigned to RDR

-- Groundspeed_Change :
--    Service_Function_Types.
--    Groundspeed_Update;
-- Destination: RDR, NAV, VIS
--
-- Assigned to RDR


--***************Function:
--*
--/ 10.25.14  Occulting


--SEND-ON-CHANGE OUTPUTS

-- Occulting_Status_Change :
--    Service_Function_Types.
--    Occulting_Status_Update;
--
-- Destination: VIS, NAV, EW, RDR
-- Assigned to VIS
```

```
--***************Function:
--*
--/ 10.25.15  Environment Support
--*

-- SEND-ON-CHANGE OUTPUTS

Malfunction_Direction_Message :
   Control_Types.
   Malfunction_Demand;
--
-- Destination: IOS
--

Environment_Segment_Simulation_State_Response :
   Control_Types.
   Segment_Simulation_State_Response;
--
-- Destination : IOS
--

Environment_Segment_Training_Mode_Response :
   Control_Types.
   Segment_Training_Mode_Response;
--
-- Destination : IOS
--

Environment_Performance_Test_Response :
   Control_Types.
   Performance_Test_Response;
--
-- Destination : IOS
--

Environment_Off_Line_Diagnostic_Response :
   Control_Types.
   Off_Line_Diagnostic_Response;
--
-- Destination : IOS
--
```

```
Environment_Remote_Controlled_Diagnostic_Response :
   Control_Types.
   Remote_Controlled_Diagnostic_Response;
--
-- Destination : IOS
--

Environment_On_Line_Diagnostic_Response :
   Control_Types.
   On_Line_Diagnostic_Response;
--
-- Destination : IOS
--

Environment_Scoring_Response :
   Control_Types.
   Scoring_Response;
--
-- Destination : IOS
--

--*
end Environment_Output_Interface;
```

APPENDIX B

Frame Load Balancing

MODULE FRAME LOAD BALANCING

Introduction. This appendix defines the frame load balancing requirements which are applicable to the Generic MSS modules.

General Module Frame Load Balancing Requirements. The following paragraphs define the frame load balancing and timing requirements for each module in the __________ *(insert application aircraft)* MSS. The maximum iteration rate for data communicated via the MSS virtual network shall be 50 Hz. This results in a frame duration of 20 ms. There shall be 16 frames per cycle in the Generic MSS. Each output message for each module shall be provided to the MSS virtual network at the assigned rate and starting with the frame defined. This shall be accomplished in accordance with paragraph 3.1.5.1.4, Synchronization and Timing, of Volume I of __________ *(insert application aircraft)* MSS System Segment Specification. For the __________ *(insert application aircraft)* MSS, the iteration rates shall be as follows:

a. ax_Rate = __ Hz

b. Half_Rate = __ Hz

c. Quarter_Rate = __ Hz

d. Eighth_Rate = __ Hz

e. Sixteenth_Rate = __ Hz

f. Send_On_Change

Send on change data shall be sent upon change of the data regardless of frame. All other messages shall be sent in the frames identified for that message as defined in the following paragraphs.

The format used to define each message is described as follows:

Message - Name of message as defined in Appendix A of this Interface Design Document (IDD).

Starting Frame - This is the first frame in which the message shall be transmitted. The message would then be retransmitted with new data at the specified rate. For example, if a half-rate message has a starting frame of 2, then it would be transmitted in frames 2, 4, 6, 8, 10, 12, 14, and 16. Frame Start Offset - This provides the ability to specify an offset or special timing characteristic for each message if required or to make one message dent on the reception of another message. For example, if the fran offset is 1.5 ms, then the message shall not be sent any sooner t ms into the frame.

Flight Station Modul Load Balancing Requirements. The following frame load balancing rements shall apply to the Flight Station module.

a. Message - Electrical_System_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Electrical_System_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Message - Hydraulic_System_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - Hydraulic_System_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

e. Message - Fuel_Management_System_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

f. Message - Fuel_Management System_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

g. Message - Pneumatic_System_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

h. Message - Oxygen_System_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

i. Message - Crew_Station_Interface_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

j. Message - Navigation_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

k. Message - Propulsion_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

l. Message - Flight_Controls_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

m. Message - EW_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

n. Message - Radar_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

o. Message - Weapons_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

p. Message - Flight_Dynamics_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

q. Message - Physical_Cues_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___

r. Message - Visual_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

s. Message - IOS_AI_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - ___ ms

Flight Controls Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Flight Controls module.

a. Message - Primary_Controls_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Misc_Control_Devices_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - Trim_Max_Rate_Outputs
Starting Frame -TBD
Frame Start Offset - __ ms

d. Message - Toe_Brakes_and_Anti_Skid_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

e. Message - AFCS_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

f. Message - Flight_Controls_Support_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Flight Dynamics Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Flight Dynamics module.

a. Message - Equations_of_Motion_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Equations of_Motion_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - Weight_and_Balance_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - Forces_and_Moments_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Propulsion Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Propulsion module.

a. Message - ine_Inlet_System_Quarter_Rate_Outputs
Starting F - TBD
Frame Star fset - __ ms

b. Message - e_Engine_Half_Rate_Outputs
Starting F - TBD
Frame Start ffset - __ ms

c. Message - Thrust_Generation_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - Engine_Bleed_Air_System_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

e. Message - Transmission_System_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

f. Message - Auxiliary_Power_Unit_System_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

g. Message - Engine_Fuel_System_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

h. Message - Engine_Exhaust_System_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

i. Message - Engine_Oil_System_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

j. Message - Propulsion_Support_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Navigation/Communication Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Navigation/Communication module.

a. Message - AHRS_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - INS_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - INS_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - INS_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

e. Message - Radar_Alt_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

f. Message - ILS_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

g. Message - TACAN_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

h. Message - TACAN_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

i. Message - UHF_VHF_HF_Intercom_Eighth_Rate_Outputs
Staring Frame - TBD
Frame Start Offset - __ ms

j. Message - IFF_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

k. Message - ADS_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

l. Message - ADS_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

m. Messare - Navigation_Support_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset

n. Message - Command_Steering_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

o. Message - HUD_Symbology_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Weapons Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Weapons module.

a. Message - Ownship_Fire_Control_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Ownship_Weapon_Dynamics_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - HUD_Max_Rate_Output
Starting Frame - TBD
Frame Start Offset - __ ms

Radar Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Radar module.

a. Message - Image_Generation_Moving_Modules_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Image_Generation_Moving_Modules_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - Mission_Computer_Interface_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - Radar_Aircraft_Systems_Interface_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Electronic Warfare Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Electronic Warfare module.

a. Message - Ownship_Chaff_and_Flares_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Ownship_Chaff_and_Flares_Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - Ownship_ECM_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - Pods_and_Controls_Eighth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

Physical Cues Module Frame Load Balancing Requirements. The Physical Cues module provides only send on change outputs to the MSS virtual network. Therefore, there are no frame load balancing requirements for the Physical Cues module.

Visual Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Visual module.

a. Message - Visual_Aircraft_Systems_Interface_Eight_Rate_Output
Starting Frame - TBD
Frame Start Offset - __ ms

Instructor/Operator Station Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the Instructor/Operator Station module.

a. Message - Clock_Tick_Message_Max_Rate
Starting Frame - TBD
Frame Start Offset - __ ms

Tactical and Natural Environments Module Frame Load Balancing Requirements. The following frame load balancing requirements shall apply to the TNE module.

a. Message - Atmosphere_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

b. Message - Weather_Quarter_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

c. Message - Ownship_Height_Above_Terrain_Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

d. Message - Moving_Models_Height_Above_Terrain_
Max_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

e. Message - Threat_Weapon_Dynamics_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

f. Message - Threat_Platform_Dynamics_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

g. Message - Companion_Vehicles_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

h. Message - External_Chaff_And_Flares_Half_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms

i. Message - External_Chaff_ And_Flares_
Sixteenth_Rate_Outputs
Starting Frame - TBD
Frame Start Offset - __ ms